Study of modelling, simulation and different operation modes and control for packed batch distillation columns

Cesur, Serap

Award date:
1993

Awarding institution:
University of Bath

STUDY OF MODELLING, SIMULATION AND DIFFERENT OPERATION MODES AND CONTROL FOR PACKED BATCH DISTILLATION COLUMNS

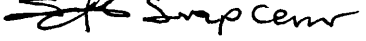
submitted by

Serap CESUR MSc.

For the degree of PhD

of the University of Bath

1993



ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

SUMMARY

This research is concerned with the development of a rigourous mathematical model to simulate multipurpose, multicomponent batch distillation in a packed column. A simulation package "PACBACDIS" was developed based on the mathematical model to study different features of the packed batch distillation. The model can deal with cases with no holdup, constant holdup and variable holdup in the packed section. Condenser holdup and pressure drop across the packed section were considered. Physical and thermodynamic properties of fluids in the column were calculated as a function of time. The model equations are a set of partial, ordinary differential and algebraic equations. These were solved by two techniques in PACBACDIS. The first is based on a finite difference scheme and PACBACDIS uses a package called DASSL (a Differential/Algebraic System Solver, (developed by Petzold, 1986)) as a tool to solve these differential algebraic equations. DASSL uses variable order and time stepping. This package uses Newton's method for solving the resultant nonlinear equations from discretization. Newton's method converges most rapidly when the initial guesses are accurate, otherwise the package fails to converge. To eliminate this defect, additional routines were written for PACBACDIS which is the second technique. This is also based on a finite difference method but uses fixed time steps and employs Gauss elimination with pivot selection for solving the system equations. Both techniques are shown to give identical results.

The model was verified by an experimental study of the operation of a batch distillation system. A laboratory column packed with No 15 Intalox metal tower packing and a binary (cyclohexane-toluene) and a ternary (cyclohexane-n.heptane-toluene) system were used. A program called DISTCONT was developed to automate the experimental

operation and a PI overhead composition controller for the experimental column was designed. The mathematical models were tested for (1) constant reflux ratio, (2) sudden reflux changes in the reflux ratio during the distillation and (3) constant overhead composition with varying reflux ratio. Application of these operational test conditions demonstrated that predictions of the model compare well with experimental results. These tests included the behaviour of the column under steady state and dynamic conditions and therefore the model can be said to simulate the whole nature of the system.

Different modes of operation (e.g., constant reflux, constant overhead composition, optimum reflux and recycled waste-cut) were studied and their modeling methods investigated.

PACBACDIS, coded in Fortran 77, has been designed to be a comprehensive simulation program which can handle a variety of problems in multicomponent, packed batch distillation operating under different modes. Beside simulation, PACBACDIS can also accomplish optimization and control of columns. The package allows the user to choose the task (simulation using various models, optimization or control), the operating policy and the model (plug flow or the superimposed axial dispersion with holdup or no holdup).

PACBACDIS has a great potential because of its inherent flexibility to all possible modes of operation and it may be extended for the selection of mode of operation and control for other mixtures and packings.

ACKNOWLEDGEMENT

I would like to take this opportunity to thank everyone who in one way or the other has helped me with my research project. In particular, I sincerely wish to thank my supervisor Dr. P L YUE for all the guidance and encouragement he has given me throughout this work and for always letting me make my own decisions and take my own initiatives.

I want to express my appreciation of the considerable help given by the technical staff of the School of Chemical Engineering, especially by J BISHOP and M WILKES. I would like to express my sorrow not to be able to take the chief technician T WALTON with me. I will always miss his humour.

I also would like to thank Prof B CRITTENDEN for his help during the design stage of my experimental apparatus. I am very grateful to Dr M GREAVES for his valuable advice on process control applications.

I also wish to thank my employer, EGE University in Turkey, for financial support to my study leave in England. I would like to express my gratitude to my beloved "hocam", Asc. Prof. B KURYEL, for his long-distance and long-lasting emotional support and encouragement. My interest in modelling was first stimulated by his lectures.

Finally, I should like to thank all my flat mates, ex- and current-. They have not only put up with me but also make great contributions to my work. Their names will not be forgotten: Nusret, Shunji, Atsuko, Christophe, Simon and Recep.

CONTENTS

LIST OF APPENDICES

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

a	....	effective area of the packing, [m^2/m^3]
AR	....	amplitude ratio
b_{Li}	....	molar liquid phase holdup of component i, [$kmol/m^3$]
b_{Vi}	....	molar vapour phase holdup of component i, [$kmol/m^3$]
B	....	molar control of the boiler at time t, [$kmol/m^3$]
B^o	....	initial molar charge to the boiler, [$kmol/m^3$]
B_L	....	total molar liquid phase holdup, [$kmol/m^3$]
B_V	....	total molar vapour phase holdup, [$kmol/m^3$]
C_L	....	total concentration of the liquid phase, [$kmol/m^3$]
C_V	....	total concentration of the vapour phase, [$kmol/m^3$]
D	....	molar rate of flow of distillate, [kmol/hr]
D_L, D_{LR}	....	dispersion coefficients of the liquid phase in the axial and radial directions, [m^2/hr]
D_V, D_{VR}	....	dispersion coefficients of the vapour phase in the axial and radial directions, [m^2/hr]
F	....	packing factor, [1/m]
g_i	....	molar flow rate of i in vapour phase, [$kmol/hr\text{-}m^2$]
G_B	....	reboiler molar holdup, [kmol]
G_c	....	condenser molar holdup, [kmol]
H	....	enthalpy, [kJ/kmol]
h	....	holdup, [m^3/m^3]
K_V	....	overall mass transfer coefficient referred to the vapour phase and multiplied by the cross-sectional area of the column, [kmol/hr atm]

K	....	equilibrium coefficient
Kp	....	static gain
k'_{yi}	....	vapour phase mass transfer coefficient for component i, [kmol/hr-m^2]
k'_{xi}	....	liquid phase mass transfer coefficient for component i, [kmol/hr-m^2]
L	....	molar flow rate of the liquid phase, [kmol/hr]
L'	....	total superficial molar flow rate of liquid phase [kmol/hr-m^2]
l_i	....	molar flow rate of i in liquid phase, [kmol/hr-m^2]
M_L	....	dynamic volumetric holdup of the liquid phase referred to a unit height, [m^3/m]
M_V	....	dynamic holdup of the vapour phase referred to a unit height, [m^3/m]
N_i	....	molar flux of i across the vapour-liquid interface, [kmol/hr-interfacial area m^2]
NC / nc	....	number of components
NJ	....	number of nodes taken in the packed section
P	....	pressure, [atm]
Q_B	....	heat supplied to the boiler, [kJ/hr]
Q_C	....	heat removed from the condenser, [kJ/hr]
Q_p	....	heat losses, [kJ/hr]
r	....	radial distance, [m]
R	....	reflux level
t	....	time, [hr]
T	....	temperature, [C]
U	....	overall heat transfer coefficient multiplied by cross sectional area of the column, [kJ/hr-C]
V	....	molar rate of flow of vapour phase, [kmol/hr]

V'	....	total superficial molar flow rate of gas phase, [kmol/hr-m^2]
v_i	....	molar flow rate of i in vapour phase, [kmol/hr-m^2]
w	....	weighting factor
W	....	frequency
X	....	mol fraction of liquid phase
x_i	....	bulk liquid mol fraction of component i
x_i^*	....	interfacial liquid mol fraction of component i
Y	....	mol fraction of vapour phase
Y^*	....	concentration of vapour phase in equilibrium with X
y_i	....	bulk vapour mol fraction of component i
y_i^*	....	interfacial vapour mol fraction of component i
z	....	axial distance, [m]
Z	....	height of column, [m]

Greek letters

α	....	relative volatility
ΔP	....	pressure drop, [mm Water]
Δt	....	increment in time, [hr]
Δz	....	spatial increment in axial direction, [m]
ε	....	dry packing void fraction, [m^3/m^3]
λ	....	latent heat of vaporization, [kJ/kmol]
ρ	....	density, [kmol/m^3]
τ_p	....	time constant

ϕ .... phase lag

Subscripts

B .... referring to the boiler

c .... referring to the condenser

D .... referring to the distillate

E .... referring to variables that enter to the packed section

i .... component index

j .... spatial index

k .... time index

L .... referring to the liquid phase

S .... referring to variables that leave from the packed section

V .... referring to the vapour phase

0 .... referring to the boiler-column interface

Z .... referring to the column-condenser interface

Superscripts

m .... iteration index

Acronyms

DASSL .... A Differential & Algebraic Systems Solver
(developed by Petzold, 1986)

DISTCONT .... Distillation Control Program
(developed in this work to automate the experimental operation)

PACBACDIS .... Packed Batch Distillation Simulation Program
(developed in this work)

SLE .... Sherwood-Leva-Eckert correlation chart for pressure drops

MOC .... Maximum Operational Capacity

CHAPTER 1

INTRODUCTION

1.1 Background

This work is concerned with the modelling, simulation and optimization of multicomponent packed batch distillation columns. It was intended to develop an experimentally verified rigourous model that would be an efficient base for studying the modes of operation and control of packed batch distillation columns.

The published work on distillation calculations has been primarily concerned with continuous distillation because of a wider industrial interest and less severe computational problems of steady state continuous distillation models. However, in recent years, the increasing consideration given to fine chemicals and the need to improve the recovery of useful resources from waste products have resulted in a renewed interest in batch distillation.

In comparison to numerous work on batch distillation in plate columns, few papers have been published on batch distillation using packed columns, because it requires the solution of systems of partial differential equations. Published work on packed batch distillation could be divided into two main groups according to their contributions into the area. The first group (Fellah et.al., (1982) and Aly et.al., (1987 and 1990)) investigate the effects of axial and radial dispersion superimposed on a plug flow model and the computational efficiency of a finite element method application to the problem. They excluded heat transfer equations from their model during the simulation because of increasing difficulty of the numerical procedure. Their model was experimentally

verified using only a binary system. The second group (Hitch and Rousseau, (1988)) implemented a finite difference method for the solution of their plug flow model, which included heat transfer equations, but the model was not experimentally verified.

1.2 Batch Distillation in General

Batch distillation is a method of separating a mixture of volatile materials into its various components.

Over the years, considerable modifications and improvements have been made in the design and operation of batch distillation. In this day of continuous processing and large scale operations, batch distillation remains an important separation technique in the manufacture of small volume, high value speciality chemicals. The decision to use a batch unit for separation depends on comparative economics between continuous and batch operation. The most outstanding attribute of batch distillation is flexibility. It has the advantage of being able to produce a number of products from a single column. The batch unit requires the least amount of capital for separating relatively pure components from a multicomponent mixture. Very little change may be required when switching from one feed mixture to another. Reflux rate and throughput can be varied easily without the need of adjusting other flows in the total process. In a situation where the composition of the feed may change frequently or where completely different mixtures must be handled from day to day, the versatility of the batch still is excellent. For multicomponent, continuous separation, separate stills or side streams are required for each component split. Batch distillation merely needs a single still and to switch receivers.

Some important aspects of batch distillation can be listed as follows:

- An unsteady-state process: The compositions and flow rates are time variant. The analysis of batch distillation is complex in comparison to continuous distillation.

- Popularity: Market demand for fragrances, pharmaceuticals and fine chemicals has been steadily growing. This increasing interest have stimulated the popularity of batch distillation.

- Applicability: Batch distillation is important and preferable to continuous distillation in the following conditions:

  - small quantities of material and high-value products,
  - irregularly scheduled periods,
  - varying feed material and initial composition,
  - different product specification.

- Flexibility: The ability of batch distillation is to produce several products from a single column. One column can be used to separate a multicomponent mixture instead of requiring NC-1 continuous columns.

Like all batch processes batch distillation has to follow an order of operation. Although some stages may be redundant or new ones can be added, stages of a typical batch distillation process are given in Table 1.1.

Table 1.1 The Successive Stages of a Typical Batch Distillation

Stage	Name	Description
1	Charging	The boiler is charged with the feed mixture.
2	Heating	Heat is applied to the boiler. The vapour works its way up the column and is condensed in the condenser. The still is allowed to come close the equilibrium under total reflux.
3	Product Drawoff	When the composition of the lightest component in the overhead distillate reaches its specification purity, a distillate product is withdrawn at a desired rate.
4	Slop cut	There is a period of time when the composition of the material being distilled becomes less rich in the most volatile component than desired specifications. This material cannot be used as the next heavier product since the second more volatile component composition does not meet its specification. Therefore a slop cut must be withdrawn until the overhead composition builds up to its specifications.
5	Cycle	Multiple products can be made from a single column, i.e., stages 3-4 are repeated. The number of product drawoff periods depends on the number of components in the original mixture charged to the kettle. The size of the slop cut recycled to the next charge depends on the sharpness of separation between the products and the sharpness of separation depends on holdup, relative volatility, reflux and number of heat transfer units.
6	Cleaning	Residues consisting of heavy ends are left in the boiler for the next batch. After several batches, however, the residues must be drawn off for redistillation or disposal.

1.3 Packed Columns

Packed columns for gas-liquid contacting are also used for distillation columns. Usually, the columns are filled with randomly oriented packing material, although in some cases the packing may be carefully positioned. The packed column is characteristically operated with counterflow of the phases.

There are various types of commercially available packings, each possessing specific advantages for liquid-gas contacting from the aspects of cost, surface availability, interface regeneration, pressure drop, weight and corrosion resistance. The packed bed is usually formed by dumping packing elements into the column and allowing them to form a random arrangement. Alternatively, larger sections of rigid, or arranged, packing material may be inserted carefully into the column.

Packed columns are usually specified when plate devices would not be feasible because of undesirable fluid characteristics or some special design requirements. Conditions favouring packed columns are:

1. For columns less than 0.6 m diameter, packings are usually cheaper than plates unless alloy metal packings are required.

2. Acids and many other corrosive materials can be handled in packed columns since construction can be of ceramic, carbon or other resistant materials.

3. Packings often exhibit desirable efficiency pressure drop characteristics for critical vacuum distillations.

4. Liquid tending to foam may be handled more readily in packed columns because of the relatively low degree of liquid agitation by the gas.

5. Holdup of liquid can be quite low in packed columns, an advantage when the liquid is thermal sensitive.

1.4 Objectives

The objectives of this thesis can be divided into two parts. The first part deals with the development of a rigourous model to simulate multipurpose, multicomponent batch distillation in a packed column. The model can deal with cases with no holdup, constant holdup and variable holdup.

PACBACDIS was developed to simulate packed batch distillation operation modes. PACBACDIS solves the model equations by two techniques. The first of these is based on a finite difference scheme and PACBACDIS uses a package as a tool called DASSL (A Differential/Algebraic System Solver, (Petzold, 1986)). DASSL can only solve an ordinary differential and algebraic equation set and therefore partial differential equations of the system were discretized to obtain their ordinary differential equation forms prior to using the package. DASSL uses variable order and time stepping. The second technique is also based on a finite difference method using fixed time steps and was developed for the cases when the first technique fails to converge.

A laboratory column packed with No 15 Intalox metal tower packing was used for an experimental study. A program called DISTCONT was developed to automate the experimental operation. A binary (cyclohexane-toluene) and a ternary (cyclohexane-n.heptane-toluene) system were used in the experimental study.

The purpose of the experimental study was to obtain experimental data for the operation of a batch distillation system that could then be used to verify the mathematical models. The mathematical models were tested under the following operation conditions: (1) constant reflux ratio, (2) sudden reflux changes in the reflux ratio during the distillation and (3) constant overhead composition with varying reflux ratio.

PACBACDIS is then used for selecting the operating mode and control for packed batch distillation columns in the second part. The results for different modes of operation (e.g., constant reflux, constant overhead composition, optimum reflux: maximum distillate, minimum batch time and maximum profit, and recycled waste-cut) are discussed. The program may be used for recommending the selection of mode of operation and control for other mixtures and packings.

In summary, this study could be divided into four following steps for the investigation of a multicomponent packed batch distillation system:

- Modelling multipurpose, multicomponent packed batch distillation columns
- Developing a simulation program for the different operation modes, PACBACDIS
- Designing a constant overhead composition PI controller for the laboratory column, DISTCONT
- Verifying the model by using the experimental results obtained from the laboratory column for a binary and a ternary system.

1.5 The Organisation of the Thesis

The structure of this thesis is as follows:

Chapter 2 covers a literature survey on modelling and simulation of batch distillation columns. The model developed for packed batch distillation columns is discussed and the simulation procedure and the simulation program "PACBACDIS" are described.

Chapter 3 includes the experimental work. The control program "DISTCONT" is introduced and the details of an overhead composition controller design are given.

Chapter 4 deals with the verification of the model developed in Chapter 2. Model and simulation parameters are investigated. The comparison of modelling and experimental results with a binary and a ternary system for a wide range of operation conditions is illustrated.

Chapter 5 is an introduction to operation modes of the batch distillation columns. Initially, an extensive literature survey is covered and then the modelling features of these modes are investigated.

Chapter 6 outlines the overall conclusions and the future work.

CHAPTER 2

DYNAMIC MODELLING AND SIMULATION OF MULTICOMPONENT BATCH DISTILLATION

2.1 Literature Review

2.1.1 Introduction

Most important processes in the chemical industry include two major operations: reaction and separation. Distillation is the most common separation operation and can either be performed continuously or in batches.

The published work on distillation calculations has been primarily concerned with continuous distillation. There are two main reasons for this. First, there is a wider industrial interest in continuous distillation; and second, unsteady state continuous distillation models are more easily derived and the related computational problems are less severe than those of batch distillation. However, in recent years, the increasing demand for speciality chemicals and the need to improve the recovery of useful resources from waste products have led to a renewed interest in batch distillation.

There are substantial differences between continuous and batch distillation processes. Unsteady state operation of continuous distillation columns usually involves relatively small deviations in composition from the steady state operation, whereas unsteady composition inherently exists in batch distillation and no steady state condition is possible. In addition, in batch distillation individual components can completely

disappear from the column, first from the reboiler and then from the entire column. Because of these two differences, the simulation work for batch distillation becomes more difficult.

Methods for the design and/or simulation of packed columns are less well developed than for plate or staged separation devices. The primary reason for this is the added complication of dealing with the rates of heat and mass transfer between the phases. King (1980) effectively describes this added complication when he states that the separation in a "discretely staged equilibrium device" is determined by equilibrium conditions alone, whereas for a continuous-contact process, mass transfer rate effects are controlling.

Of all the numerical solution procedures available for the design or simulation of multicomponent separation processes, by far the most rigourous and efficient algorithms are those that deal with stagewise mass-transfer operations.

2.1.2 Plate Type Batch Distillation Column Models

Multicomponent batch distillation models consist of a system of differential-algebraic equations that describe the mass and heat balances of the process and the vapor-liquid equilibria between phases. The models can be differentiated into three categories: rigourous, semi rigourous and short-cut models.

The essential feature of a rigourous model is that it incorporates complete mass and heat accumulation in all the holdups, mass and heat transfer from the system to the surroundings, and variation of relative volatility of the components with the operating conditions. Although a semi-rigourous model is a simplification of a rigourous model,

there is no real boundary between the rigourous and semi-rigourous models. The rigourous model is directed more toward the accuracy of the model, while the semi rigourous model gives more consideration to computational time.

A short-cut model is by far the simplest to describe the material balance of the system by approximation or simplifications based on experimental considerations. It tries to approximate the relationship between the instantaneous composition of the liquid in the accumulator with that of the liquid in the still without involving tray to tray calculations. Therefore, such models usually neglect the heat balance, the tray and the accumulator holdups and the variation of the relative volatility of the components with temperature and pressure.

2.1.2.1 Rigourous Model of Multicomponent Batch Distillation

The first comprehensive model for multicomponent batch distillation was published by Meadows (1963). The model employed heat and material balances as well as volume balances and was limited only by the assumption of ideal trays, constant volume tray and accumulator holdup, adiabatic operation and negligible vapour holdup. Meadows solved the model using a finite difference method starting from the steady state total reflux condition. It was found that the use of a finite difference method for solving the model leads to instability if too large a step size was used, hence long computational times were required. He suggested that future work should be developed for a more refined method to increase the speed of calculations. He also recommended the development of a start-up procedure that would allow heat and time requirements to be calculated from the point of first application of heat to the still.

Distefano (1968) devoted his study mainly to the analysis of various methods for numerical integration of the differential equations comprising a comprehensive mathematical model of multicomponent batch distillation. He used the same assumptions as Meadows made in his model and started the solution from the steady state total reflux condition. Further, Distefano compared ten numerical methods and concluded that the Adams Moulton Shell Predicator-corrector method showed the most favourable stability characteristics. He also concluded that even the most stable numerical integration techniques then available were not entirely satisfactory for the solution of transient distillation equations and other so-called "stiff" differential equations.

Distefano's method of solution has been tested experimentally by Steward et.al. (1973) who proved that the experimental and theoretical results were in good agreement. Therefore, the Distefano method (1968) could be used as a comparison for any further development in the method of solution.

The work of Distefano (1968) was followed by Boston et.al. (1981) who developed an advance system known as the BATCHFRAC model for the simulation of batch distillation operations. They enhanced Distefano's (1968) model by introducing multiple feed and side product streams. There were no improvements to the method of solution, and the start-up procedure remained untouched.

Guy (1983) developed a complete model similar to that of Boston et.al. (1981). He used the concepts of constant liquid molar holdup, hydraulic time constant and real tray hydraulics as variations to the constant liquid volume holdup assumption. He also introduced a Murphree tray efficiency instead of the assumption of ideal trays.

Sodotomo's model (1983) assumed a constant molar holdup, a negligible vapour holdup and generated a stable, rapid procedure. This procedure turns the problem into one of finding the eigenvalues of a symmetrical tri-diagonal matrix by performing a similarities transformation on the coefficient matrix. This eigenvalue procedure is more than twenty times faster than the Runge-Kutta-Gill method on the offered example. Although Sodotomo showed that the procedure was stable and fast, the constant molar holdup assumption is a limitation to his model because of the changes in compositions on each plate.

Jang (1990) developed a rigourous numerical procedure for the simulation of batch distillation and solved it using Gear's method. The mathematical model was formulated on the assumptions that vapour holdups are negligible, the effluent streams are in thermodynamic equilibrium, the column pressure remains constant throughout the time period, the drawoff fluid dynamic lags are negligible, the column operates adiabaticaly and the tray is theoretical. Finally, instead of constant molar or mass liquid holdup, the more realistic assumption that the constant volume of liquid holdups is used.

Diwekar and Madhavan (1991) developed a simulation package, BATCH-DIST, for the design, simulation and optimization of multicomponent, multifraction batch distillation columns operating under different modes. The package includes simulation models of varying degrees of complexity and rigour; efficient but simplified models (based on short-cut methods) for preliminary design and rapid analysis of column behaviour, and rigourous models for verification and detailed design. Besides simulation and design, BATCH-DIST can also accomplish optimization and optimal control of columns. It was extensively tested for both binary and multicomponent systems, ideal as well as nonideal systems, and columns with and without holdup effects.

2.1.2.2 Semi-Rigourous Model of Multicomponent Batch Distillation

An example of a semi-rigourous model of multicomponent batch distillation is that suggested by Luyben (1971). This was based on the assumption of constant relative volatilities, constant molar holdups in the trays and accumulator, constant vapour flow with flat profiles and it neglected the heat balance. Luyben (1971) proposed a simple procedure to simulate the startup period. All the holdups are initially charged with liquid of the same composition, it is not clear how to do it in practice, but presumably it can be done by charging the liquid from the accumulator down to the still. He suggested another alternative that the system can be assumed to have all the tray holdups initially filled with liquid having the composition in equilibrium with the composition of the liquid in the still.

Domenech and Enjalbert (1981) made a slight modification to the model of Luyben(1971). They still incorporated the variation of relative volatilities of the components with temperature and pressure. It was stated that a sub-program would be needed to start the solution from the steady state total reflux condition. However, it is also possible to start solution based on temperature and concentration profiles supplied as data. This implies that their model can be used to simulate a similar startup procedure to that suggested by Luyben(1971). They solved their model using Runge-Kutta-Merson with variable step size.

Galindez and Frendenslung (1988) solved the above model using their quasi-steady state approach. The unsteady state process of batch distillation is simulated as a succession of short periods of time in which continuous distillations are carried out. They were able to simulate the startup procedure as suggested by Luyben (1971).

Domenech and Enjalbert (1978) presented the common semi-rigourous model, that is a model without holdup, all the tray holdups and accumulator holdups are neglected. The model is composed of an overall mass balance and the component balance equation.

$$\frac{dG_B}{dt} = -\frac{V}{R+1} \tag{2.1}$$

$$\frac{dX_{i,B}}{dt} = -\frac{V}{R+1}\frac{X_{i,D} - X_{i,B}}{G_B} \tag{2.2}$$

$$Y_{i,n} = K_{i,n}X_{i,n} = \frac{R}{R+1}X_{i,n-1} + \frac{X_{i,D}}{R+1} \tag{2.3}$$

i=1,2,....,NC

n=1,....,N

$$\sum_{i=1}^{NC} K_{i,n}X_{i,n} - 1.0 = 0 \tag{2.4}$$

2.1.2.3 Short-cut Model of Multicomponent Batch Distillation

The general form of total mass balance and the component balance for a short-cut model are as follows.

$$\frac{dG_B}{dt} = -\frac{V}{R+1} \tag{2.5}$$

$$\frac{dX_{i,B}}{dt} = \frac{1}{R+1}\frac{V}{G_B}(X_{i,B} - X_{i,D}) \tag{2.6}$$

where $X_{i,D} = f_i(X_{i,B}, R, N, \alpha_i, constants)$

Diwekar and Madhavan (1986) suggested a shorcut model based on the assumption that a batch distillation can be considered as continuous distillation with changing feed. That is, for a small interval of time the batch column behaviour is analogous to a continuous column. Therefore they derived their model from the most widely used short-cut model for continuous distillation. The equations used in the proposed short-cut model are: Hengstebeck-Geddes, the Fenske, Underwood and Gilliland equations.

2.1.3 Packed Batch Distillation Column Models

In comparison with the numerous works on batch distillation in plate columns, relatively few papers have been published on batch distillation using a packed column. Although a substantial amount of work has been done recently on numerical methods for packed column design and simulation, both the quantity and quality of available packed column algorithms still rate far below the levels achieved in staged column simulation. Deficiencies remain evident in several key areas, one major problem that has yet to be addressed adequately is the inability of current numerical schemes to solve, without the threat of encountering convergence difficulties, the complicated systems of nonlinear model equations required for a rigourous treatment of simultaneous mass and heat transfer phenomena. Another significant deficiency is the lack of flexibility of most current packed column algorithms, i.e. they are typically capable of performing calculations on only one type of separation process under a relatively narrow range of operating conditions. Other aspects of packed column simulation that have not been considered in a rigourous way include (1) unsteady state operations of continuous processes and (2) batch processes such as batch distillation.

The modeling of multiphase transfer in many unit operations requires the solution of systems of partial differential equations. Because of the complexity of the mechanisms that are involved, the equations are usually nonlinear and rarely can be solved analytically. Recourse to numerical methods, based on approximations aimed at transforming the differential system to an algebraic system, is therefore necessary. This approach is needed when studying continuous or batch, packed distillations, because these models consist of partial differential equations to describe their dynamic behaviour.

It is worthwhile to examine the models used for packed batch distillation columns and to compare the performance of the various numerical techniques for the selection of the most appropriate procedure for each application.

The widely used method for designing packed distillation columns is based on the theory of height equivalent to a theoretical plate (HETP). The first proposal by Peters (1922) was really the application of the stagewise mechanism for the plate column to the packed tower. However the process is one of continuous countercurrent mass transfer. The HETP is an experimentally determined quantity, characteristic of each type of packing. Unfortunately it is found that HETP varies, not only with the type and size of the packing but also very strongly with the flow rates of each fluid and concentration as well, so that an enormous amount of experimental data would have to be accumulated to permit utilization of the method. The difficulty lies in the failure to account for the fundamentally different action of tray and packed towers, and the HETP method has now largely been abandoned.

The most general model is that proposed by Fellah (1982), which was based on and improved the work of Lee (1976). This deals with the dynamic simulation of batch

distillation and considers the nonlinearity of the vapor-liquid equilibria, axial dispersion and the energy balances. An experimental study on a pilot plant allowed evaluation of the various models. Problems encountered in integrating the equations by a finite difference method were noted, in particular in ensuring rapid and stable convergence. This study was initiated by Aly (1985). He included radial dispersion in the model equations and tried to improve the stability of the convergence by using a finite element method of the Galerkin-type.

The general equations of the model that were presented by Fellah (1982) and Aly (1985) distinguish between the packed section of the column of height Z and radius R_0 and its boundaries with the boiler and the condenser. The packed section consists of Δz elements. Figure 2.1 illustrates mass transfer terms for a Δz element and heat transfer terms are shown in Figure 2.2. Mass and heat transfer equations given for a Δz element in the packed section are as follows:

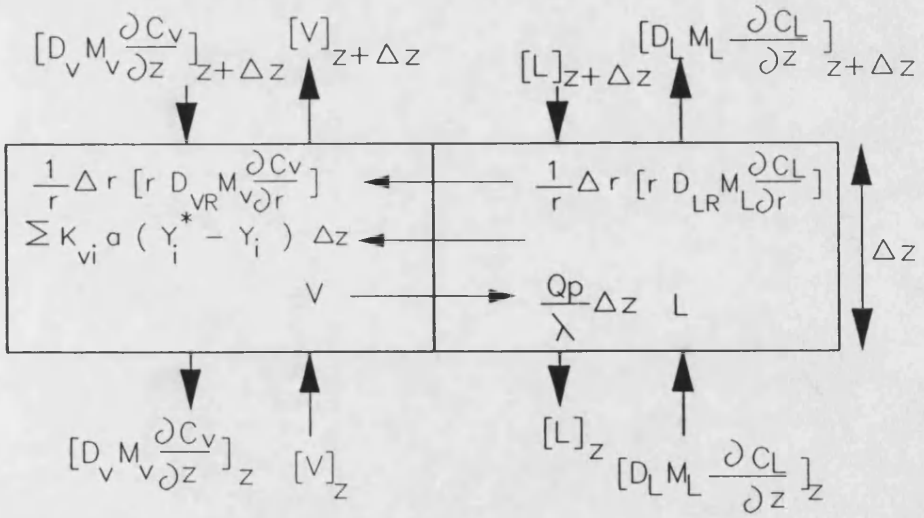


Figure 2.1 Mass Transfer in Δz element

The equations of the vapour phase for component i:

$$\frac{\partial(M_v C_v y_i)}{\partial t} = -\frac{\partial(V y_i)}{\partial z} + \frac{\partial}{\partial z}\left[D_V M_v \frac{\partial(C_v y_i)}{\partial z} \right] + \frac{1}{r}\frac{\partial}{\partial r}\left[r D_{vR} M_v \frac{\partial(C_v y_i)}{\partial r} \right]$$

$$+ K_{vi} a (y_i^* - y_i) - \frac{Q_p}{\lambda} x_i^* \qquad (2.7)$$

where

$\frac{\partial(M_v C_v y_i)}{\partial t}$ = accumulation term

$\frac{\partial(V y_i)}{\partial z}$ = plug flow term

$\frac{\partial}{\partial z}\left[D_V M_v \frac{\partial(C_v y_i)}{\partial z} \right]$ = axial dispersion term

$\frac{1}{r}\frac{\partial}{\partial r}\left[r D_{vR} M_v \frac{\partial(C_v y_i)}{\partial r} \right]$ = radial dispersion term

$K_{vi} a (y_i^* - y_i)$ = the mass transfer between liquid and vapour phases

$\frac{Q_p}{\lambda} x_i^*$ = vapour condensation because of heat loss

Total mass transfer equation:

$$\frac{\partial(M_v C_v)}{\partial t} = -\frac{\partial V}{\partial z} + \frac{\partial}{\partial z}\left[D_V M_v \frac{\partial C_v}{\partial z}\right] + \frac{1}{r}\frac{\partial}{\partial r}\left[r D_{vR} M_v \frac{\partial C_v}{\partial r}\right]$$

$$+ \sum_i K_{vi} a (y_i^* - y_i) - \frac{Q_p}{\lambda} \qquad (2.8)$$

where $\sum_i K_{vi} a (y_i^* - y_i) = 0$ for binary systems

Heat transfer equation:

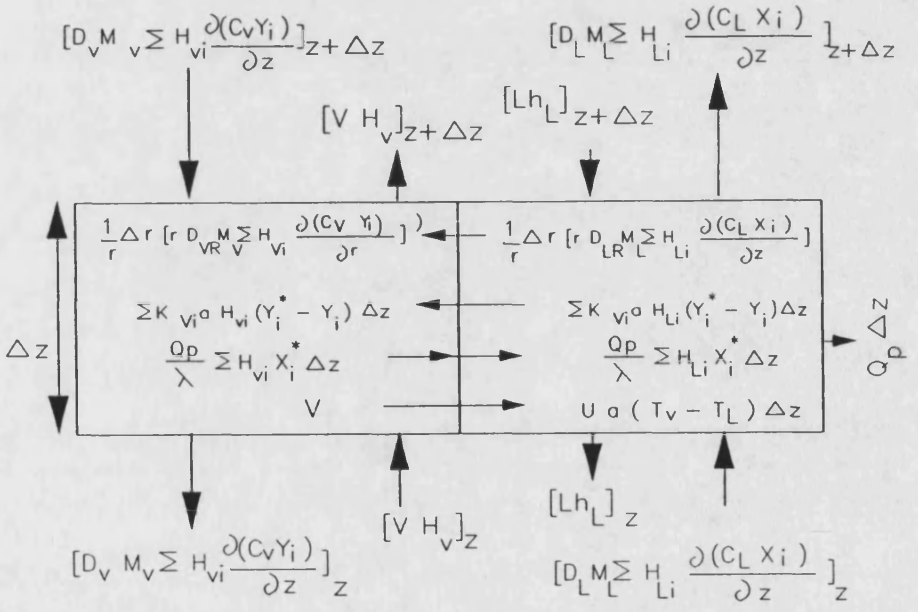


Figure 2.2 Heat Transfer in Δz element

$$\frac{\partial(M_v C_v H_v)}{\partial t} = -\frac{\partial(VH_v)}{\partial z} + \frac{\partial}{\partial z}\left[D_V M_v \sum_i H_{vi} \frac{\partial(C_v y_i)}{\partial z}\right]$$
$$+\frac{1}{r}\frac{\partial}{\partial r}\left[rD_{vR} M_v \sum_i H_{vi} \frac{\partial(C_v y_i)}{\partial r}\right] + \sum_i K_{vi} a H_{vi}(y_i^* - y_i)$$
$$-\frac{Q_p}{\lambda}\sum_i H_{vi} x_i^* - Ua(T_v - T_l) \qquad (2.9)$$

The equations for the liquid phase are:

$$\frac{\partial(M_L C_L x_i)}{\partial t} = \frac{\partial(Lx_i)}{\partial z} + \frac{\partial}{\partial z}\left[D_L M_L \frac{\partial(C_L x_i)}{\partial z}\right] + \frac{1}{r}\frac{\partial}{\partial r}\left[rD_{LR} M_L \frac{\partial(C_L x_i)}{\partial r}\right]$$
$$-K_{vi} a(y_i^* - y_i) + \frac{Q_p}{\lambda} x_i^* \qquad (2.10)$$

Total mass transfer equation:

$$\frac{\partial(M_L C_L)}{\partial t} = \frac{\partial L}{\partial z} + \frac{\partial}{\partial z}\left[D_L M_L \frac{\partial C_L}{\partial z}\right] + \frac{1}{r}\frac{\partial}{\partial r}\left[rD_{LR} M_L \frac{\partial C_L}{\partial r}\right]$$
$$-\sum_i K_{vi} a(y_i^* - y_i) + \frac{Q_p}{\lambda} \qquad (2.11)$$

Heat transfer equation:

$$\frac{\partial(M_L C_L H_L)}{\partial t} = \frac{\partial(LH_L)}{\partial z} + \frac{\partial}{\partial z}\left[D_L M_L \sum_i H_{Li} \frac{\partial(C_L X_i)}{\partial z}\right]$$
$$+\frac{1}{r}\frac{\partial}{\partial r}\left[rD_{LR} M_L \sum_i H_{Li} \frac{\partial(C_L X_i)}{\partial r}\right] - \sum_i K_{vi} a H_{vi}(y_i^* - y_i)$$
$$+\frac{Q_p}{\lambda}\sum_i H_{Li} x_i^* + Ua(T_v - T_l) - Q_p \qquad (2.12)$$

where

a = gas-liquid interfacial area per unit volume of packing, area/volume [m^2/m^3]

C_L, C_V = total concentrations of liquid and vapour phases [$kmol/m^3$]

M_L, M_V = dynamic volumetric holdups in the height of an element for the liquid and vapour phase, [m^3/m]

L, V = liquid and vapour phase molar flow rates [kmol/s]

D_L,D_{LR},D_V,D_{VR} = dispersion coefficients in axial and radial directions of liquid and gas phases, [m/s]

K_V = overall mass transfer coefficient for vapour phase multiplied by cross sectional area of the column, [kmol/s atm]

U = overall heat transfer coefficient multiplied by cross sectional area of the column, [cal/s C]

Aly et.al. (1987) deal with the modeling of batch, packed distillation and the numerical solution of the equations for steady state operation with total reflux when radial dispersion neglected. The model took into account plug flow of the phases with axial dispersion and therefore involved only one independent variable, axial spatial term. He also presented a numerical treatment of the algorithm when both axial and radial dispersions are superimposed as well as an analysis of the dynamic behaviour associated with the withdrawal of distillate.

The operating conditions were as follows:

- The operating pressure is assumed to be constant and the pressure drop to be negligible.
- The boiler functions as a theoretical plate with the vapour and liquid in thermodynamic equilibrium.
- The vapours are totally condensed, the reflux is introduced into the column, and the distillate is removed at its boiling point, i.e, a total condenser.
- No condenser holdup

Although an almost complete model was presented, the authors had to make the following simplifying hypotheses to ease the structure of the model and to simplify the numerical treatment by the Galerkin method:

- Thermal losses are negligible, the column is adiabatic.
- The physical and thermodynamic properties are constant along the length of the column and in time.
- The molar fluxes of the liquid and vapour are constant. This eliminates heat balance equations.
- The number of independent variables has been reduced by dealing with a binary system (cyclohexane-toluene), however, it was stated that, the algorithms can be easily extended for complex mixtures.

In this case, the model equations were reduced to the partial mass balance equations only for liquid and vapour phases in the packed section, the supplementary reboiler and condenser equations and the boundary conditions at the limits of the packed section.

Aly et.al. (1987) showed that the choice of a Galerkin-type method of finite element integration leads to a much more complex and difficult implementation than with finite difference. The numerical implementation for the steady state led to results that are essentially identical for both of the two methods of integration, the finite difference and finite element methods. However, a more rapid convergence can be observed for a finite element method, which allows a coarser discretization of the domain of integration (and hence fewer elements).

When axial and radial dispersion are superimposed on plug flow, two spatial variables, z and r are required. Because of the complexity of the model, the search for reliable algorithms and the reduction of the volume and duration of the calculations are as important as precise modeling of physical phenomena. Pibouleau (1984) used a finite difference method to integrate a model for steady state operation of a packed distillation column with axial and radial dispersion.

A study (Aly (1985)) of the sensitivity of the model to variations of the coefficients of radial dispersion D_{VR} and D_{LR}, as well as of the effect of the radius R_o of the column on the radial composition profile was also carried out.

Aly (1987) reached a conclusion from the collection of numerical tests that were carried out for distillation in a packed column: the terms in the model for the radial dispersion

have marginal importance. Therefore, these terms have been eliminated in the model for dynamic operation of the column. On the other hand, the axial dispersion phenomenon is always indispensable for accurately representing the column operation.

Hitch and Rousseau (1988) developed an algorithm, named DISTCAL, to handle the inherently unsteady-state problem of batch distillation involving narrow-boiling mixtures. He stated that, prior to their work, no rigourous numerical procedure for the simulation of batch distillation in a packed column had ever been presented in the scientific literature. The details of the formulation of the model equations and the mathematical structure of the numerical algorithm, DISTCAL, are discussed below. Simulation predictions were evaluated qualitatively by a parametric study to test the validity of the mathematical model and the numerical solution procedure. The following assumptions have been made in their model:

- The column is adiabatic, i.e., no heat loss from the column
- The pressure drop through the column is negligible
- Gas and liquid streams are in plug flow, i.e., there are no radial temperature or concentration gradients and no axial dispersion

The following model is introduced in their work:

Unsteady-state individual component material balances

$$\frac{\partial b_{Li}}{\partial t}+\frac{\partial b_{Vi}}{\partial t}=\frac{\partial l_i}{\partial z}-\frac{\partial v_i}{\partial z} \qquad (2.13)$$

mass transfer relationships

a. Liquid side

$$\frac{\partial b_{Li}}{\partial t} = \frac{\partial l_i}{\partial z} + k'_{x,i} a (x_i^* - x_i) \tag{2.14}$$

b. Vapour side

$$\frac{\partial b_{Vi}}{\partial t} = -\frac{\partial v_i}{\partial z} - k'_{y,i} a (y_i - y_i^*) \tag{2.15}$$

An analysis of the degree of freedom of this system readily shows that with these differential equations alone the problem is underspecified. To obtain a fully determined system, the variables appearing in these balance equations must also satisfy the following equations.

The individual component liquid and vapour molar holdups

$$b_{Li} = B_L x_i \tag{2.16}$$

$$b_{Vi} = B_V y_i \tag{2.17}$$

The individual component liquid and vapour molar flow rates

$$l_i = L' x_i \tag{2.18}$$

$$v_i = V' y_i \tag{2.19}$$

equilibrium relationships

$$y_i^* = K_i x_i^* \tag{2.20}$$

unsteady-state, adiabatic energy balance

$$\frac{\partial(B_L H_L)}{\partial t} = \frac{\partial(L' H_L)}{\partial z} - \frac{\partial(V' H_V)}{\partial z} \tag{2.21}$$

unsteady-state, total material balance

$$\frac{\partial B_L}{\partial t} + \frac{\partial B_V}{\partial t} = \frac{\partial L'}{\partial z} - \frac{\partial V'}{\partial z} \tag{2.22}$$

summation equation

$$\sum_{i=1}^{nc} y_i^* = \sum_{i=1}^{nc} K_i x_i^* = 1 \tag{2.23}$$

Hitch and Rousseau stressed the point that the commonly adopted assumption of negligible gas phase holdup was not made when deriving the unsteady state material balances; it will be shown in Chapter 4 that this term plays a significant role in describing the time dependent behaviour of the experimental system chosen for simulation.

The numerical method chosen for the unsteady state simulation uses finite difference expressions to approximate the derivatives in the differential model equations, thereby reducing the system of partial differential equations to a coupled system of algebraic equations. Hitch (1986) reduced the first order spatial derivative terms by forward and backward difference approximations for the liquid and vapour streams. These are respectively

$$\frac{\partial l_i}{\partial z} = \frac{l_i(J+1) - l_i(J)}{\Delta z} \tag{2.24}$$

$$\frac{\partial v_i}{\partial z} = \frac{v_i(J) - v_i(J-1)}{\Delta z} \tag{2.25}$$

A solution of this set of finite difference equations is obtained at each of several grid points separated by a uniform grid size. In the above equations, the points J+1, J and J-1 refer to three interior mesh points in the finite difference grid containing NJ points.

A similar expression was used to approximate the time dependent derivative terms, this is analogous to the forward difference approximation for spatial derivatives. For example, the use of finite difference formula to approximate both time and spatial derivatives in the unsteady state material balance for component 1,

$$\frac{b_{Ll}(J,N+1)-b_{Ll}(J,N)}{\Delta t}+\frac{b_{Vl}(J,N+1)-b_{Vl}(J,N)}{\Delta t}=$$

$$\frac{l_1(J+1,N+1)-l_1(J,N+1)}{\Delta z}-\frac{v_1(J,N+1)-v_1(J-1,N+1)}{\Delta z} \qquad (2.26)$$

This implicit scheme, also known as the Backward Euler method, generates a system of algebraic equations that may be written as follows

$$\sum_{k=1}^{n}(A_{i,k}(J,N)C_k(J-1,N+1)+B_{i,k}(J,N)C_k(J,N+1)$$

$$+D_{i,k}(J,N)C_k(J+1,N+1))=G_i(J,N) \qquad (2.27)$$

i=1,2,....,n equation number

k=1,2,....,n variable number

These algebraic difference equations with appropriate expressions for the boundary conditions at J=1 and J=NJ may be written in a block tridiagonal matrix form as

$$\begin{bmatrix} B(1,N) & 0 & 0 & 0 & 0 \\ A(2,N) & B(2,N) & D(2,N) & 0 & 0 \\ . & . & . & . & . \\ 0 & A(J,N) & B(J,N) & D(J,N) & 0 \\ . & . & . & . & . \\ 0 & 0 & 0 & 0 & B(NJ,N) \end{bmatrix} \begin{bmatrix} C(1,N+1) \\ C(2,N+1) \\ . \\ C(J,N+1) \\ . \\ C(NJ,N+1) \end{bmatrix} = \begin{bmatrix} G(1,N) \\ G(2,N) \\ . \\ G(J,N) \\ . \\ G(NJ,N) \end{bmatrix}$$

or $\quad M \times T = Z \qquad (2.28)$

where each of the terms in the coefficient matrix, M, is itself an NV x NV submatrix, NV being the number of dependent variables in the system of model equations. Similarly, each term in the solution vector, T, is itself a vector of dimension NV as is each term in the right hand side vector, Z.

This matrix has to be decomposed once for each time step to achieve an approximation to the actual individual component concentration profiles over the entire column at that point in time. The accuracy of the solution, at any given time, depends on the magnitudes of both Δt and Δz; smaller step sizes give more accurate solution. This is the major drawback of the finite difference approach, because a large number of points is frequently required, this can result in an excessive computational time requirement.

The preferred method for testing the validity of a simulation procedure is to compare the predictions of the algorithm with actual experimental results. Because of the nonavailability of experimental data for comparison with simulation results, Hitch (1986) conducted a parametric study by systematically varying the input specifications to the simulation followed by qualitatively evaluating the effects that these variations had on algorithm predictions.

Thus, faced with the absence of useful data in the literature dealing with batch distillation in packed columns, Hitch and Rousseau (1988) studied the influence of height of packing, boilup rate, reflux ratio and condenser holdup had on algorithm predictions.

2.1.4 Conclusion

Publications on the modelling and simulation of multicomponent packed batch distillation are still rare. It could be concluded from Table 2.1 that there is a need for further development of a dynamic rigourous model that will consider liquid and vapour packed section holdups, condenser holdup, pressure drop and a complete heat balance for variable flow rates.

Hitch (1988) stated that there is still considerable room in the area of packed column simulation. The primary area is the promotion of computational efficiency, i.e., the ability of a simulation to arrive at an accurate solution with a minimum of computational time and effort.

Unfortunately, a natural conflict exists between the need for rigourous yet stable algorithms and the desire for computational efficiency. Fundamental problems such as this must usually be solved through the implementation of some sort of compromise. The present research is an attempt to address some of the problems highlighted above.

Table 2.1 Review on Packed Batch Distillation Modelling Works

Author	**Model Specifications** (Equations Developed)	**Simulation Specifications** (Assumptions made in their solution)	**Experimentally Verified and Experimental Data published**
Fellah et.al. (1982)	multicomponent non-adiabatic axial dispersion liquid and vapour holdup in packed no condenser-reflux drum holdup	adiabatic column binary system equimolal flow rates constant physical and thermodynamic properties constant pressure no pressure drop finite central difference method	Experimentally verified with a binary mixture (Cyclohexane - Toluene) No experimental data published
Aly et.al. (1987)	multicomponent non-adiabatic radial and axial dispersion holdup in packed no condenser holdup	adiabatic equimolal flow rates negligible pressure drop constant physical and thermodynamic properties finite element method (Galerkin type)	Experimentally verified with the same binary system data used by Fellah et.al (1982)
Aly et.al. (1987) in French (1990) translated into English	multicomponent non-adiabatic axial dispersion holdup in packed no condenser holdup	adiabatic equimolal flow rates negligible pressure drop constant physical and thermodynamic properties finite element and finite difference methods were compared	Experimentally verified with the same binary system data used by Fellah et.al (1982)
Hitch and Rousseau (1988)	multicomponent adiabatic plug flow model variable holdup condenser holdup	ternary system nonideal variable flow rates variable physical and thermodynamic properties finite difference (constant step)	No
Cesur (1993) This work	multicomponent adiabatic axial dispersion variable holdup condenser holdup	multicomponent nonideal variable flow rates variable physical and thermodynamic properties pressure drop finite difference with variable step size	Yes with a binary (Cyclohexane - Toluene) and a ternary (Cyclohexane - n-Heptane - Toluene) systems Experimental data available

2.2 The Present Dynamic Model

This section shows development a dynamic model of multicomponent packed batch distillation. The model has to satisfy certain conditions to extend the study suggested by Aly et al.(1987) and Hitch and Rousseau (1988). The condition demands that the model has to be more rigourous so that a startup period before the steady state total reflux condition is achieved and a complete heat balance of the system can be taken into account.

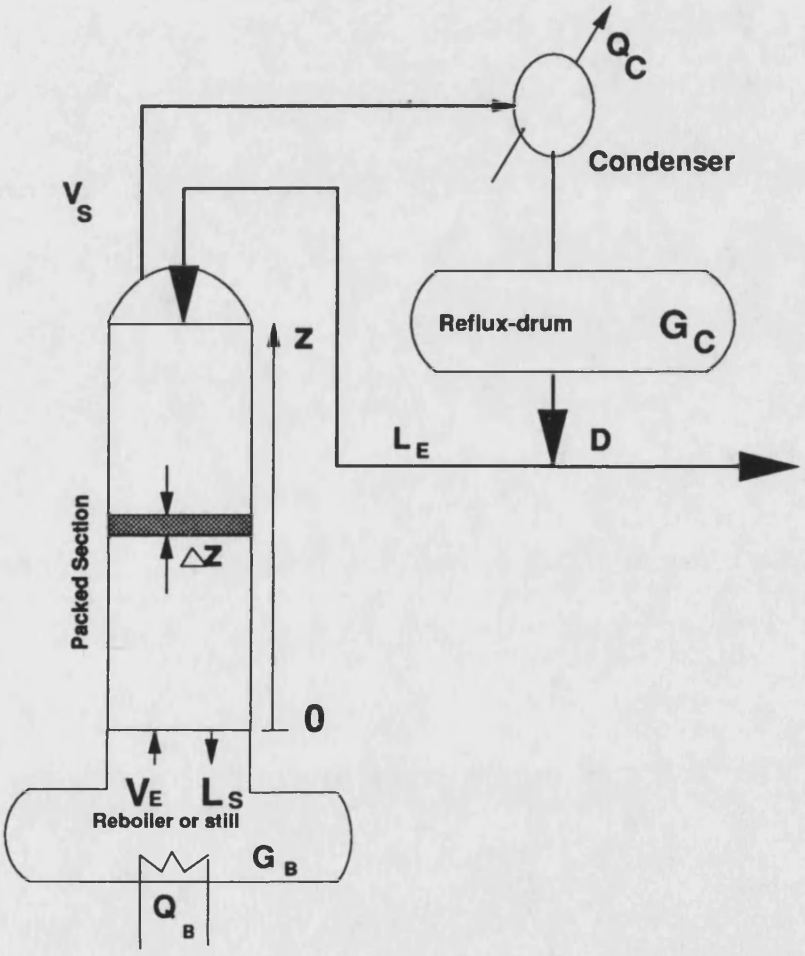


Figure 2.3 Sketch of a packed batch distillation column

2.2.1 Dynamic Model Equations

Assumptions are made to achieve a satisfactory model and equations are derived under the following conditions:

- Thermal losses are negligible and the column is adiabatic.
- Removal of the heat transfer relationships is based on the assumption that, at any vertical position in the column, the bulk liquid and gas phase temperatures will be very close to the interfacial temperature.
- The column is equipped with a total condenser.
- The boiler functions as a theoretical plate with the vapour and liquid in thermodynamic equilibrium.
- The physical and thermodynamic properties change along the length of the column and in time.
- Axial dispersion is superimposed on plug flow and radial dispersion is neglected. Therefore there are no radial temperature or concentration gradients. This condition may become invalid for large, randomly packed columns. It may, however, be used appropriately in structured packings, despite the column diameter.
- A multicomponent and nonideal system with variable gas and liquid holdups in packed section.

- During the distillation process the pressure at top of the column is assumed constant with respect to time. This assumption was confirmed by the experimental results. Pressure drop across the column is calculated from pressure drop correlation for each time step. A linear pressure profile in the packed section is considered.

- Variable condenser holdup is considered.

Packed Section:

The changes in concentration with height in packed towers are continuous rather than stepwise as for tray towers, and the computation procedure must take this into consideration.

The packed section is considered as a complete unit and principally, it is divided into hypothetical stages with Δz height in which perfect mixing is assumed. The continuous change in concentration is thus replaced by a series of steps that may be chosen sufficiently small to represent the original continuous profiles well. Equations written for a Δz infinitesimal element in this section are as follows.

The transfer of mass in the packed column takes place throughout the column at the interface between the liquid and vapour. Both liquid and vapour phases are moving generally countercurrently, with the liquid making its way through the maze of passages in and around the packing to the bottom of the column and the vapour rising through the same passages to the top of the column. The rate of mass transfer in equimolar counterdiffusion can be expressed as

$$N_A = - N_B$$

for binary systems. For multicomponent systems,

$$\sum_{i=1}^{n} N_i = 0$$

where n is the number of components.

Since the more volatile component, A is diffusing from the liquid phase into the vapour phase, there must be a concentration gradient in the direction of mass transfer within each phase. This can be shown graphically as in Figure 2.4, where a section through the two phases in contact is shown. Even where the usual simplifying assumptions are not strictly applicable, within a section of the column V and L are both sufficiently constant for equimolar counterdiffusion between phases to be practically true. Consequently, mass transfer coefficients are $F_V=k'_y$ and $F_L=k'_x$ for vapour and liquid phases respectively, and the mass transfer flux is

$$N_A = \frac{d(Vy)}{a\,dZ} = k'_y(y^* - y) = \frac{d(Lx)}{a\,dZ} = k'_x(x - x^*)$$

$$-k'_{xi}a(x_i - x_i^*) + k'_{yi}a(y_i^* - y_i) = 0 \qquad (2.29)$$

For packed towers, rates of flow are based on unit tower cross-sectional area, mol/(area)(time).

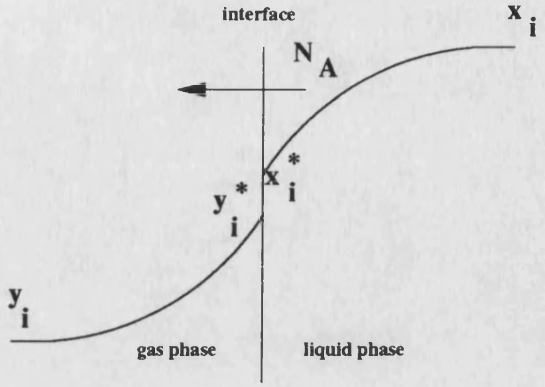


Figure 2.4 A concentration gradient in the direction of mass transfer within each phase

Mass transfer relationships

a- Liquid phase

$$B_L \frac{\partial x_i}{\partial t} = L' \frac{\partial x_i}{\partial z} + D_L B_L \frac{\partial^2 x_i}{\partial z^2} - k'_{xi} a (x_i - x_i^*) \tag{2.30}$$

b- Vapour phase

$$B_V \frac{\partial y_i}{\partial t} = -V' \frac{\partial y_i}{\partial z} + D_V B_V \frac{\partial^2 y_i}{\partial z^2} + k'_{yi} a (y_i^* - y_i) \tag{2.31}$$

$$y_i^* = K_i x_i^* \tag{2.32}$$

i=1,....,NC NC:Number of Components

Unsteady-state, adiabatic energy balance

$$\frac{\partial(B_V H_V)}{\partial t}+\frac{\partial(B_L H_L)}{\partial t}=\frac{\partial(L' H_L)}{\partial z}-\frac{\partial(V' H_V)}{\partial z}$$

$$+D_L H_L\frac{\partial^2 B_L}{\partial z^2}+D_V H_V\frac{\partial^2 B_V}{\partial z^2} \tag{2.33}$$

Unsteady-state, total material balance

$$\frac{\partial B_V}{\partial t}+\frac{\partial B_L}{\partial t}=\frac{\partial L'}{\partial z}-\frac{\partial V'}{\partial z}+D_L\frac{\partial^2 B_L}{\partial z^2}+D_V\frac{\partial^2 B_V}{\partial z^2} \tag{2.34}$$

Total condenser-reflux drum:

Total Material Balance:

$$\frac{dG_c}{dt}=V_S-L_E-D=V_S-D(R+1) \tag{2.35}$$

$$R=\frac{L_E}{D} \tag{2.36}$$

Component Material Balances:

$$\frac{d(G_c x_{i,c})}{dt}=V_S y_{i,S}-L_E x_{i,E}-D x_{i,D}$$

$$x_{i,c}=x_{i,E}=x_{i,D}$$

$$\frac{dx_{i,c}}{dt}=-\left(\frac{L_E+D+\frac{dG_c}{dt}}{G_c}\right)x_{i,c}+\left(\frac{V_S}{G_c}\right)y_{i,S}$$

$$\frac{dx_{i,c}}{dt} = \frac{V_s(y_{i,s} - x_{i,c})}{G_c} \tag{2.37}$$

The Enthalpy (Energy) Balances:

$$Q_c = V_S(H_{VS} - H_{LE}) - G_c\frac{dH_{LE}}{dt} \tag{2.38}$$

Reboiler:

Total Material Balances:

$$\frac{dG_B}{dt} = L_S - V_E \tag{2.39}$$

Component Material Balances:

$$\frac{d(G_B x_{i,B})}{dt} = L_S x_{i,S} - V_E y_{i,E}$$

$$y_{i,E} = y_i^*(x_{i,B}) \tag{2.40}$$

$$\frac{dx_{i,B}}{dt} = \frac{L_s x_{i,s} - V_E y_{i,E} - x_{i,B}(L_s - V_E)}{G_B} \tag{2.41}$$

The Enthalpy (Energy) Balances:

$$\frac{d(G_B H_{LB})}{dt} = Q_B + L_S H_{LS} - V_E H_{VE}$$

$$Q_B = V_E(H_{VB} - H_{LB}) + L_S(H_{LB} - H_{LS}) + G_B \frac{dH_{LB}}{dt} \tag{2.42}$$

or

$$V_E = \frac{Q_B + L_s(H_{Ls} - H_{LB}) - G_B \frac{dH_{LB}}{dt}}{(H_{VB} - H_{LB})} \tag{2.43}$$

Finally, the thermodynamic equilibrium equations, correlations for computing liquid and vapour holdups, the transfer and dispersion coefficients and other terms as a function of physical, thermodynamic and hydrodynamic properties and of molar fractions of liquid and vapour phases occur in the equations.

Phase equilibriums:

$$y^*_{i,j} = K_{i,j} x^*_{i,j} \tag{2.44}$$

where K_{ij} is the equilibrium constant

$$K_{i,j} = K_{i,j}(T_j, P_j, \vec{x}^*_j, \vec{y}^*_j)$$

Mole fraction sum:

$$\sum_i y^*_{i,j} = \sum_i K_{i,j} x^*_{i,j} = 1.0 \tag{2.45}$$

2.2.1.1 Boundary Conditions

The boundary conditions are the flow rates and concentrations of the vapour and liquid phases at the two ends of the column.

In the equations, the subscripts 0 and Z denote the boiler-column interface and the column-condenser interface respectively. The boundary conditions express the continuity of the outlet variables at the frontiers of the system and the absence of axial mixing outside the packed section of the column. Conditions analogous to those proposed by Danckwerts (1953) have been used:

z=0 boiler-column interface

Component Material Balances:

$$V_E y_{Ei} = V y_i - D_V B_V \frac{\partial y_i}{\partial z}\Big|_{z=0} \tag{2.46}$$

$$L_z x_{zi} = L x_i + D_L B_L \frac{\partial x_i}{\partial z}\Big|_{z=0} \tag{2.47}$$

$$\frac{\partial (B_L x_i)}{\partial z}\Big|_{z=0} = 0 \tag{2.48}$$

or

$$\frac{\partial (L x_i)}{\partial z}\Big|_{z=0} = 0 \tag{2.49}$$

Total Material Balance:

$$V_E = V - D_V \frac{\partial B_V}{\partial z}\Big|_{z=0}$$

$$\frac{\partial L}{\partial z}\Big|_{z=0} = 0$$

Energy Balance:

$$V_E H_{VE} = V H_V - D_V \sum_i H_{Vi} \frac{\partial (B_V y_i)}{\partial z}\Big|_{z=0}$$

$$\frac{\partial (L H_L)}{\partial z}\Big|_{z=0} = 0$$

z=Z column-condenser interface

Component Material Balances:

$$L_E x_{Ei} = L x_i + D_L B_L \frac{\partial x_i}{\partial z}\Big|_{z=Z} \qquad (2.50)$$

$$\frac{\partial (B_V y_i)}{\partial z}\Big|_{z=Z} = 0 \qquad (2.51)$$

Total Material Balance:

$$L_E = L + D_L \frac{\partial B_L}{\partial z}\Big|_{z=Z}$$

$$\frac{\partial V}{\partial z}\Big|_{z=Z} = 0$$

Energy Balance:

$$L_E H_{LE} = | LH_L + D_L \sum_i H_{Li} \frac{\partial(B_L x_i)}{\partial z} |_{z=Z}$$

$$| \frac{\partial(VH_V)}{\partial z} |_{z=Z} = 0$$

2.2.1.2 Holdups

The dynamic model considers holdup in three main parts of the system, namely in the reboiler, G_B [kmol liquid holdup], in the condenser-reflux drum, G_C [kmol liquid holdup] and the packed section, B_L and B_V [kmol liquid or vapour holdup respectively/m^3 of column].

in the condenser-reflux drum:

$$G_C = M_C \rho_C \tag{2.52}$$

in the packed section:

The packed section holdups can be described by one of the following three models. PACBACDIS considers all three models. Each holdup model defines a different equation set for the packed section.

1. No holdup model, B_L and $B_V = 0$
2. Constant volume-variable molar holdup model

   h_{Lt} and h_{Vt} are constant volume holdups and molar holdups can be calculated by

   $$B_L = h_{Lt} \rho_L \tag{2.53}$$

   $$B_V = h_{Vt} \rho_V \tag{2.54}$$
3. Variable volume holdup model

Correlations for both static (h_{Ls}) and operating (h_{Lo}) liquid holdups as functions of packing type, size, flow rates and the physical properties of the liquid are given by Treybal (1980), in Table 6.5. The total liquid holdup (h_{Lt}) is:

$$h_{Lt} = h_{Lo} + h_{Ls} \tag{2.55}$$

The total volumetric vapour holdup (h_{Vt}) may then be assumed to be the difference between the dry packing void fraction (ε) and the total liquid holdup; i.e.,

$$h_{Vt} = \varepsilon - h_{Lt} \tag{2.56}$$

The value of ε for several packing types and sizes may be found in Treybal (1980) in Table 6.3 and Coulson et al.(Vol.2, 1978) in Table 4.3 and for new type of packings, data obtained from Norton Company is presented in Appendix 1.

Manufacturer's data are often available on holdup for specific packings but not for new types of packings. Therefore, Leva (1953)'s liquid holdup equation is used for approximate calculations for the conditions below the loading point.

$$h_{Lt} = 0.143\left(\frac{L'}{d}\right)^{0.6} \tag{2.57}$$

where L'= the liquid flow rate in [kg/m^2 s]

d = the equivalent diameter of the packing in [mm] that is the diameter of a sphere with the same surface to volume ratio as a packing particle=$\frac{6(1-\varepsilon)}{a_p}$

h = the holdup in [m^3 of liquid of vapour phase/m^3 of column]

Once the total volumetric liquid and vapour holdups are known, the corresponding total molar holdups (B_L and B_G) are found through the relationships (2.53) and (2.54).

and in the reboiler:

$$G_B = G_B^o - \sum_{j=0}^{NJ}(G_{Lj} + G_{Vj}) - G_c - \int_0^t D\,dt \qquad (2.58)$$

G = The molar holdup [kmol]

M = The volume holdup [m^3]

G_B^o = The initial molar charge to the reboiler [kmol]

ρ = molar density [$kmol/m^3$]

2.2.1.3 Pressure Drop in the Packed Section

For several decades, the Sherwood-Leva-Eckert (SLE) correlation chart has been the standard of the industry for predicting flooding points and pressure drops for random packing. Perry's Handbook, 6th Ed. (1984) recommends one of Eckert's versions of the chart as the sole correlation to use for flooding and pressure drop prediction for random packings.

The acceptance of the SLE chart has sharply declined during the past couple of decades. This prompted the appearance of a large number of alternative correlations. Most of these entirely abandoned the SLE chart, while others retained a few of the features of the charts. Many attempted to substitute a more fundamental predictive model for the empiricism in the SLE chart. Although many of these new correlations have shown

promising results, for many reasons none has gained acceptance by industry-wide. Many are based on a limited body of data. Many only apply to a restricted range of packing geometries and have not been tested for other geometries.

Eckert (1975) produced a later version of the SLE correlation chart than that given in Perry's Handbook. The later version omitted the flood curve and also made some other minor adjustments. Strigle (1987) changed the scales of Eckert's later version of the SLE correlation from log-log to semilog to make interpolation between adjacent pressure drop curves easier. Strigle also updated packing factors based on Eckert's later version. Kister and Gill (1991) stated that both from their current analysis and Strigle's clear demonstration, overall the SLE correlation gives good pressure drop predictions.

A generalized pressure drop correlation revised by Norton Company has been used in this work. This generalized pressure drop correlation, given in Appendix 2, uses a constant packing factor for each packing in the entire range of abscissa values.

The abscissa of the correlation is known as the flow parameter:

$$X = \frac{L}{G}\sqrt{\frac{\rho_G}{\rho_L}}$$

The ordinate is the capacity parameter:

$$Y = \frac{CG^2 F \nu^{0.1}}{\rho_G(\rho_L - \rho_G)}$$

F is the packing factor, which is an empirical value characteristic of the packing size and shape, [ft^{-1}]. F replaces the term [a/ε^3], where a is the specific surface area per unit

volume of bed [ft^2/ft^3] and ε is the fractional voidage of bed of packing. Values of the packing factor are available in both open and manufacturers' literature. Packing factors are given for all commonly used dumped packings in Appendix 1. To use these packing factors with either metric or British system units, a term, C has been included in the ordinate.

Kister and Gill (1991) correlated the available flood pressure drop data against the relevant packing factors. The relationship was then expressed as

$$\Delta P_{flood} = 0.115F^{0.7}$$

ΔP=pressure drop, in. of water/ft of packing

This equation permits estimation of the flooding pressure drops for all random packings. Once this pressure drop is known, the flooding velocity can be back calculated using the SLE data charts.

It is given by Norton Company that maximum operational capacity (MOC), the maximum vapour rate that provides normal efficiency of a Intalox type of packing, occurs at 95% of the flood point velocity.

Estimating flooding and pressure drop using the SLE data charts essentially involves data interpolation and extrapolation within the framework of the pressure drop correlation.

2.2.1.4 End Effects and Axial Mixing

Mass transfer will occur where the liquid is introduced by sprays or other distributors and where it drips off the lower packing support. This constitutes the end effects. To

make a proper evaluation of the packing, experimental data must be corrected for these effects, usually by operating at several packing heights and then extrapolating the results to zero height. Alternatively, equipment design with corrected data will include a small safety factor represented by the end effects. However the data represented by the equations for calculating mass transfer coefficients, holdups and interfacial areas are believed to be reasonably free of end effects (Treybal (1980)).

The equations generally used for design, which are the same as those used for computing mass transfer coefficients from laboratory measurements, are based on the assumption that the gas and liquid flow in piston, or plug flow, with a flat velocity profile across the tower and with each portion of the gas or liquid having the same residence time in the tower. This is not actually the case, nonuniformity of packing and maldistribution of liquid may lead to channelling, or regions where the liquid flow is abnormally great. Further the liquid in the static holdup moves forward much more slowly than that in the moving holdups and drops of liquid falling from a packing may be blown upward by the gas. Similarly the resistance to gas flow at the tower walls and in relatively dry regions of the packing is different from that elsewhere; the downward movement of liquid induces downward movement of the gas. As a result, the purely piston-type flow does not occur, and there is a relative movement within each fluid parallel to the axis of the tower, variously described as axial mixing, axial dispersion and back mixing. Axial dispersion is the spreading of the residence time in unidirectional flow due to the departure from purely piston-type, or plug flow, the fluid particles move forward but at different speeds. Back mixing is the backward flow in a direction opposite to that of the net flow, caused by frictional drag of one fluid upon the other, spray of liquid into the gas, and likewise.

Diffusion and mixing of the fluids flowing in packings usually have an adverse effect on the performance of such contacting devices. The normal axial concentration or temperature gradient tends to be flattened if different elements of the fluid stream have different velocities; even with macroscopic plug flow, axial diffusion and mixing will reduce concentration gradients in the direction of flow. Mean driving forces for mass and heat transfer are reduced and performance is adversely affected (Sherwood et al.(1975)).

Dispersion normal to the direction of flow (radial) is usually helpful, in that the concentration differences between small fluid elements tend to be equalized and so reduce the variations that enhance the unwanted axial dispersion.

Departures from piston or plug flow are the result of axial dispersion, caused by one or more of the following:

- a radial velocity gradient in the tower
- eddy diffusion or mixing
- molecular diffusion

It should be pointed out that the extent of mixing is much greater in the liquid phase than the gas phase and gas phase Peclet numbers decrease with both liquid and gas flow rates. On the other hand, liquid Peclet numbers increase with liquid flow rates, and no quantitative effect on gas rate is observed (Dunn et al.(1977)).

Effective diffusivity D is used to describe axial mixing that accounts for all the factors causing mixing in the axial direction, these include the transfer of material in and out of

stagnant pockets and local channelling, as well as the effects of turbulent eddy diffusion (Sater and Levenspiel (1966)). Admittedly, the use of a single term to account for these many effects may be an oversimplification, but it should describe the flow characteristics in a packed bed better than the plug flow model.

In spite of the large amount of data now available, as yet no general correlation has been made that may be used with confidence to predict either D_{ax} or D_r.

2.2.1.5 Model Parameters

To solve the model equations, certain physical and thermodynamic properties must be estimated. For example, the individual fluid mass transfer coefficients (k_x, k_y) and the interfacial area a are necessary and they are functions of fluid properties, flow rates and the type of packing used.

The methods used for model parameter calculations are listed in Table 2.2. The parameters change along the column with changing temperature, pressure and composition, and can be expressed in general form as follows, such as for liquid and vapour enthalpies:

$$h_{L,j} = h_{L,j}(T_j, P_j, \vec{x}_j)$$

$$H_{V,j} = H_{V,j}(T_j, P_j, \vec{y}_j)$$

Table 2.2 Model Parameters

1. Gas Viscosity

   •Pure component-Chapman and Enskog theory in Reid et al. 3th Edition (1977), page 394

   •mixture viscosity-Dean and Steil (1965) in Reid et al. 3th Edition (1977), page 419

2. Liquid Viscosity

   Reid et al. 3th Edition (1977), page 629

3. Gas Density

   Ideal gas law

4. Liquid Density

   Yen and Woods (1966) in Reid et al. 3th Edition (1977), page 61

5. Binary Gas Diffusivity

   Wilke-Lee Modification to Chapman-Enskog relationship (1955)

   Reid et al. 3th Edition (1977), p.555

6. Binary Liquid Diffusivity

   Scheibel correlation (1954)

   Reid et al. 3th Edition (1977), p.571

   Perry's Handbook, 6th Edition (1984), p.3-287

7. Gas-Phase Mixture Diffusivity

   Wilke (1950)

8. Liquid-Phase Mixture Diffusivity

   Wilke (1950)

9. Mass Transfer Coefficients and Interfacial Area

   Onda, et al.(1968)

10. Equilibrium Distribution Coefficients

    Soave-Redlich-Kwong equation of State

    Grayson-Streed Modification of Chao-Seader Method, Holland (1981)

11. Liquid and Vapour Phase Holdup

    Leva (1953) given in Coulson et al., Vol.2, 3th Edition, p.164

12. Ideal Gas Enthalpy

    API Method (7A1.1, Page 7-7)

13. Liquid and Gas Enthalpy

    Departure Functions from equation of state by Soave-Redlich-Kwong equation of State

14. Vapour Pressure

    Frost-Kalkwarf-Thodds in Reid et al. 3th Edition (1977), p.188

15. Liquid Surface Tension

    Hakim et al., in Reid et al. 3th Edition (1977), p.608

16. Liquid Heat Capacity

    Sternling-Brown, in Reid et al. 3th Edition (1977), p.156

17. Liquid Thermal Conductivity

    Weber (1880) equation in Coulson et al., Vol.6, p.244

18. Liquid Phase Axial Dispersion Coefficients

    Kushalkar and Pangarkar (1990)

    Sater and Levenspiel (1966)

    Dunn et al. (1977)

    Co and Bibaud (1971)

19. Vapour Phase Axial Dispersion Coefficients

    Sater and Levenspiel (1966)

    Dunn et al. (1977)

2.2.2 Steady-State Model Equations

For steady-state behaviour with total reflux, the model equations are obtained by excluding the time dependent terms.

Packed Section:

$$-k'_{xi}a(x_i - x_i^*) + k'_{yi}a(K_i x_i^* - y_i) = 0 \tag{2.59}$$

$$L'\frac{\partial x_i}{\partial z} + D_L B_L \frac{\partial^2 x_i}{\partial z^2} - k'_{xi}a(x_i - x_i^*) = 0 \tag{2.60}$$

$$-V'\frac{\partial y_i}{\partial z} + D_V B_V \frac{\partial^2 y_i}{\partial z^2} + k'_{yi}a(K_i x_i^* - y_i) = 0 \tag{2.61}$$

$$y_i^* = K_i x_i^* \tag{2.62}$$

$$\frac{\partial L'}{\partial z}-\frac{\partial V'}{\partial z}+D_L\frac{\partial^2 B_L}{\partial z^2}+D_V\frac{\partial^2 B_V}{\partial z^2}=0 \tag{2.63}$$

$$\frac{\partial (L'H_L)}{\partial z}-\frac{\partial (V'HV)}{\partial z}+D_L H_L\frac{\partial^2 B_L}{\partial z^2}+D_V H_V\frac{\partial^2 B_V}{\partial z^2}=0 \tag{2.64}$$

Condenser:

$$L_E = V_S \tag{2.65}$$

$$x_{E,i} = y_{S,i} \tag{2.66}$$

$$Q_C = V_S(H_{VS} - H_{LE}) \tag{2.67}$$

Reboiler:

$$y_{E,i} = y^*(x_{B,i}) \tag{2.68}$$

$$x_{i,B} = \frac{G_B^o x_{i,B}^o - \sum_{j=0}^{NJ}(G_{Lj}x_{i,j} + G_{Vj}y_{i,j}) - G_C x_{i,C}}{G_B} \tag{2.69}$$

$$G_B = G_B^o - \sum_{j=0}^{NJ}(G_{Lj} + G_{Vj}) - G_c \tag{2.70}$$

$$Q_B = V_E(H_{VB} - H_{LB}) - L_S(H_{LS} - H_{LB}) \tag{2.71}$$

2.3 Simulation Procedure

The running of a packed batch distillation column depends on four main parameters:

1. the initial reboiler charge, G_B°,
2. the initial composition of that charge, x_B°,
3. the reflux ratio, R,
4. the boilup rate V or the amount of heat Q_B supplied the reboiler that determines the boilup rate.

In this work, the paired sequential method is used to linearise the system of partial differential model equations by matching each model variable with the particular model equation that most directly determines the value of the variable. The complete set of 4NC+3 variables included in this new, modified model is listed in Table 2.3.

Table 2.3 Model Variables of jth Δz Element in Packed Section

1. NC liquid bulk component mole fractions	x_j
2. NC vapour bulk component mole fractions	y_j
3. NC liquid interfacial component mole fractions	x_j^*
4. NC vapour interfacial component mole fractions	y_j^*
5. 1 liquid-phase total flow rate	L'
6. 1 vapour-phase total flow rate	V'
7. 1 interfacial temperature	T^*

Bulk and interfacial mole fractions (x_j, y_j, x_j^*, y_j^*) are paired with the individual component mass balances (eq. (2.29)), the mass transfer relationships (eq. (2.30) and eq. (2.31)) and the equilibrium expressions (eq. (2.32)).

Distillation typically involves a narrow-boiling set of constituents, so that, variations in total enthalpy will change the flow rates of each phase but will have little effect on column temperatures, which depend mostly on the composition. Therefore, it is preferable to use an energy balance (eq. (2.33)) and a total material balance (eq. (2.34)) to compute bulk liquid and vapour flow rates, while determining column temperatures via bubble-point (equilibrium) calculations by using eq. (2.45).

In batch distillation, there are essentially two modes of operation that Holland and Liapis (1983) refer to as the start-up and product periods. During the start-up period, the column is operated at total reflux until the reflux reaches a desired purity. Because no product is withdrawn, the column may achieve steady-state during the start-up period. However once the product period begins and distillate is withdrawn, unsteady-state operation of the column is inherent because of the composition of the liquid in the stillpot is changing continually. The computer program developed for this work, PACBACDIS is capable of simulating both modes of operation for batch distillation in a packed column.

The steady-state model involves ordinary differential equations in the variable z, where z is axial spatial term, and the dynamic model is made up of partial differential equations with two variables z and t, where t is time, both models also include algebraic equations.

Two approaches were used to solve the partial differential/algebraic equation set of the dynamic model. The first one was built primarily around systems of finite-difference equations that are solved implicitly using matrix decomposition techniques. Central difference and forward difference approximations were applied to the spatial and time terms respectively. Gauss elimination method was used to calculate the solution to the resulting linear equations.

The second approach for solving dynamic model equations is to discretize the spatial terms only, and thus convert the partial differential equation (PDE) system into an ordinary differential equation (ODE) initial value problem or a differential/algebraic equation (DAE) system for a current problem. The DAE system can then be solved using DASSL (A Differential/Algebraic Systems Solver Language).

2.3.1 Discretization Procedure

The equations of the model were discretized by the method of central, forward and backward finite differences. The three point backward spatial differencing was used near the reboiler and the condenser in the column, where they represent frontiers of the domain of integration where the variation of the two phases are the greatest.

2.3.1.1 Steady state simulation

for j=1,...,N-1

$$\left(\frac{\partial U}{\partial z}\right)_j = \frac{U_{j+1} - U_{j-1}}{2\Delta z}$$

$$\left(\frac{\partial^2 U}{\partial z^2}\right)_j = \frac{U_{j+1} - 2U_j + U_{j-1}}{\Delta z^2}$$

for j=0 reboiler-packed section interface

$$\left(\frac{\partial U}{\partial z}\right)_0 = \frac{-3U_0 + 4U_1 - U_2}{2\Delta z}$$

for j=N packed section-condenser interface

$$\left(\frac{\partial U}{\partial z}\right)_N = \frac{U_{N-2} - 4U_{N-1} + 3U_N}{2\Delta z}$$

2.3.1.2 Dynamic Simulation

for j=1,...,N-1

$$\left(\frac{\partial U}{\partial t}\right)_{j,k} = \frac{U_{j,k+1} - U_{j,k}}{\Delta t}$$

$$\left(\frac{\partial U}{\partial z}\right)_{j,k} = \frac{U_{j+1,k} - U_{j-1,k}}{2\Delta z}$$

$$\left(\frac{\partial^2 U}{\partial z^2}\right)_{j,k} = \frac{w}{\Delta z^2}(U_{j+1,k+1} - 2U_{j,k+1} + U_{j-1,k+1}) + \frac{1-w}{\Delta z^2}(U_{j+1,k} - 2U_{j,k} + U_{j-1,k})$$

w is the weighting factor that is proposed 1/2 by Cranck-Nicholson.

for j=0 reboiler-packed section interface

$$\left(\frac{\partial U}{\partial z}\right)_{0,k} = \frac{-3U_{0,k} + 4U_{1,k} - U_{2,k}}{2\Delta z}$$

for j=N packed section-condenser interface

$$\left(\frac{\partial U}{\partial z}\right)_{N,k} = \frac{U_{N-2,k} - 4U_{N-1,k} + 3U_{N,k}}{2\Delta z}$$

2.3.2 DASSL, A Differential / Algebraic Systems Solver

This is a program for the numerical solution of implicit systems of differential/algebraic equations (developed by L. R. Petzold, 1986 and distributed by Sandia National Laboratories) of the form

$$F(t,y,y')=0$$
$$y(t_0)=y_0 \qquad (2.72)$$
$$y'(t_0)=y_0'$$

F, y and y' are N dimensional vectors.

PACBACDIS calls DASSL routines for the solution of differential and algebraic equations of the packed batch distillation model. DASSL is useful for solving two general classes of problems that cannot be handled by standard ordinary differential equation (ODE) solvers.

1• It is not possible to solve the problem for y' explicitly while rewriting the problem as a standard form ODE system y'=f(t,y).

2• It is possible in theory to solve for y', but it is impractical to do so. For example, to convert Ay'=By to standard form, we must multiply by A^{-1}. If A is a sparse matrix, A^{-1} may not be sparse, so it is advantageous to be able to solve the equations in their original form.

Nearly all the most popular codes for solving ODE's have been directed at systems written in the standard form:

$$y'=f(t,y) \qquad (2.73)$$
$$y(t_0)=y_0$$

Several programs beside DASSL have been written for solving systems that cannot be written in standard form. In the early 1970's, Gear (1971) first noticed that numerical methods for solving stiff differential systems could be adapted to solve some DAE systems, and Gear and Brown (1973) wrote a code for this purpose.

In 1980, Soderlind published a program (DASP3) for solving systems of the form

$$y'=f(t,y,z) \tag{2.74}$$
$$y(t_0)=y_0$$
$$0=g(t,y,z)$$

Hindmarsh and Painter (1981) released a code (LSODI) for solving linearly implicit DAE systems,

$$A(y,t)y'=f(t,y) \tag{2.75}$$

LSODI is similar to DASSL in that it uses backward differentiation formulas (BDF) to advance the solution from one step to the next. However, there are substantial differences between these two codes, both in how they are used and in the strategies that are used internally to compute the solution.

2.3.2.1 How DASSL Works

The underlying idea of Gear for solving DAE systems is to replace the derivative in the problem (2.72) by a difference approximation, and then to solve the resulting equation for the solution at the current time t_n using Newton's method.

For example, the first order formula is obtained by replacing the derivative by the backward difference in (2.72),

$$F\left(t_n, y_n, \frac{y_n - y_{n-1}}{\Delta t_n}\right) = 0 \tag{2.76}$$

This equation is then solved using Newton's method,

$$y_n^{m+1} = y_n^m - \frac{F\left(t_n, y_n^m, \frac{y_n^m - y_{n-1}}{\Delta t_n}\right)}{\left(\frac{\delta F}{\delta y'} + \frac{1}{\Delta t_n}\frac{\delta F}{\delta y}\right)} \tag{2.77}$$

where m is the iteration index.

The algorithms used in DASSL are an extension of this approach. Instead of using the first order formula, DASSL approximates the derivative using the k^{th} order backward differentiation formula (BDF), where k ranges from one to five. On every step it chooses the order k and step size Δt_n, based on the behaviour of the solution.

To solve (2.72) numerically at time t_n by the k-step BDF, $y'(t_n)$ is replaced by $\rho y_n/h$ where ρ is the difference operator defined by

$$\rho y_n = \sum_{i=0}^{k} \alpha_i y_{n-i} \tag{2.78}$$

$h=t_n-t_{n-1}$ and α_i are the BDF coefficients, this gives the system of nonlinear equations

$$F\left(t_n, y_n, \frac{\rho y_n}{h}\right) = 0 \tag{2.79}$$

Newton's method converges most rapidly when the initial guess y_n^0 is accurate. DASSL obtains an initial guess for y_n by evaluating the polynomial that interpolates the computed

solution at the last k+1 times $t_{n-1}, t_{n-2}, \ldots, t_{n-(k+1)}$, at the current time t_n.

Once y_n^0 is found, Newton's method is used to solve for y_n as in eq.(2.76), except that in general the derivative is approximated by the k^{th} order BDF formula, instead of by the backward difference of y_n. When the step size is not constant, there is a choice as to the form of the BDF formula to use. DASSL uses the fixed leading coefficient form of the BDF formula. These formulas tend to be more stable than the fixed coefficient formulas used in LSODI, and are more efficient in some respects than the variable coefficient formulas used in EPISODE.

It is important to solve the nonlinear equation (2.76) efficiently. To simplify notation, this equation can be rewritten as

$$F(t, y, \hat{\alpha}y + \beta) = 0 \tag{2.80}$$

where $\hat{\alpha}$ is a constant that changes whenever the stepsize or order changes, β is a vector that depends on the solution at past times and t, y, $\hat{\alpha}$, β are evaluated at t_n. This equation is solved in DASSL by a modified version of Newton's method,

$$y^{m+1} = y^m - c\left(\frac{\delta F}{\delta y'} + \alpha\frac{\delta F}{\delta y}\right)^{-1} F(t, y^m, \hat{\alpha}y^m + \beta) \tag{2.81}$$

The iteration matrix

$$G = \frac{\delta F}{\delta y'} + \alpha\frac{\delta F}{\delta y}$$

is computed and factorised, and is then used for as many time steps as possible. In general, the value of α when G was last computed is different from the current value of $\hat{\alpha}$, (2.81) may not converge. The constant c in (2.81) is chosen so as to speed up the convergence rate when $\alpha \neq \hat{\alpha}$ and is given by

$$c = \frac{2}{\left(1 + \frac{\hat{\alpha}}{\alpha}\right)} \tag{2.82}$$

The rate of convergence ρ of (2.81) is estimated whenever two or more iterations have been taken by

$$\rho = \left(\frac{\| y^{m+1} - y^m \|}{\| y^1 - y^0 \|} \right)^{1/m} \tag{2.83}$$

The norms are scaled norms that depend on the error tolerances specified by the user. The iteration has converged when

$$\frac{\rho}{1-\rho} \| y^{m+1} - y^m \| < 0.3 \tag{2.84}$$

If $\rho > 0.9$ or $m>4$, and the iteration has not yet converged, the stepsize is reduced, and/or an iteration matrix based on current approximations to y, y' and α is formed and the step is attempted again.

The linear systems are solved using routines from the LINPACK subroutine package. The matrix can either be dense or have a banded structure. There is the facility for the user to write a routine to compute G, given t,y,y' and α. For some problems, this can be more efficient than using finite differences to compute the matrix.

After the corrector iteration has converged, an error test is made to determine whether the solution satisfies a local error tolerance specified by the user. The test is satisfied whenever

$$c \parallel y_n - y_n^0 \parallel \leq \lambda$$

where c is a constant that depends on the order and recent stepsize history of the method. The constant c is chosen to control both the variable stepsize local truncation error and the error in interpolated values of y between mesh points. If the error test is satisfied, the code takes another step. Otherwise, the stepsize and/or order are reduced and the step is attempted again.

The stepsize and order for the next step are determined using the following strategies: The code estimates what the error would have been if the last few steps had been taken at constant step size, at the current order k, and at k-2, k-1, and k+1. If these estimates increase as k increases, the order is lowered; if they decrease, it is raised. The new stepsize Δt_{n+1} is chosen so that the error estimate based on taking constant stepsizes Δt_{n+1} at order k_{n+1} satisfies the error test.

One of the main complications that arise in solving DAE's, which has no counterpart in ODE's, is that it may not be a trivial matter to obtain initial values for all the components of y or y'. Depending upon the application, users may know y_0 but not y_0', y_0' but not y_0, or various combinations of these possibilities. It is fairly common for users to know all y_0, and some but not all the components of y_0'. To facilitate solving problems of this type, there is an option in DASSL to compute the initial values for y', given the initial

values of y and an initial guess for y_0'. The algorithm uses the backward Euler method in conjunction with a damped Newton iteration. A stepsize Δt_0 is chosen and the iteration matrix is computed at

$$y = y_0 + \Delta t_0 y'_0$$

$$y' = y'_0$$

This algorithm works best when the iteration matrix does not depend on y', or depends on this value only weakly.

The code is arranged so that driver routine called DASSL, allocates storage, checks for illegal input and other error conditions, sets up the initial stepsize and optionally calls a subroutine to compute the initial derivative. Communication between DASSL and the other routines is via parameter lists and one labelled common block whose elements can only be altered by the driver.

2.3.2.2 Using DASSL

The most important information the code needs is how to define the function F in (2.72), which describes the equation to be solved. To define F, the user writes a subroutine RES that takes as input the time T and the vectors Y and YPRIME, and produces as output the vector DELTA, where DELTA=F(T,Y,YPRIME) is the amount by which the function F fails to be zero for the input values of T, Y and YPRIME. The subroutine has the form

```
SUBROUTINE RES(T,Y,YPRIME,DELTA,IRES,RPAR,IPAR)
```

The parameter IRES is an integer flag that DASSL always sets to zero prior to calling RES. It is used to flag situations where an illegal value of Y or a stop condition has been encountered. For example, in some applications, if a component of Y becomes even slightly negative, then the function F cannot be evaluated. In this case, the user would check for a negative component of Y upon entering RES. If one is found, then the user would set IRES=-1 and return without evaluating the function. DASSL then cuts the stepsize and attempts the step again. RPAR and IPAR are real and integer vectors, respectively, and are at the user's disposal for communication purposes. They are never altered by DASSL or any of its subroutines.

To get started, DASSL needs a consistent set of initial values T,Y and YPRIME. This means that we must have F(T,Y,YPRIME)=0 at the initial time. As was pointed out earlier, finding a consistent set of initial values for a given problem may not be trivial, and there is an option in DASSL to compute the initial value for YPRIME.

The call of the function DASSL is of the term:

```
CALL DASSL    (RES,NEQ,T,Y,YPRIME,TOUT,INFO,RTOL,ATOL,
               IDID,RWORK,LRW,IWORK,LIW,RPAR,IPAR,JAC)
```

The parameters are described in detail in the documentation of DASSL (Petzold, 1982).

For a few problems in applications whose analytic solutions are always positive, it was found that, it is crucial to avoid small negative solutions that can arise due to truncation and roundoff errors. There is an option in DASSL to enforce nonnegativity of the solution.

The relative and absolute error tolerances RTOL and ATOL can be specified either as vectors or as scalars. Most of the important decisions in the code make use of these tolerances to compute weights for the norm, and the finite difference Jacobian approximation makes use of them directly when there is no other information available about the scale of the solution. The selection of these tolerances accurately reflects the scale of the problem. For users who are not familiar with how to set the tolerances RTOL and ATOL, it is recommended to start with the following rule of thumb. Let m be the number of significant digits required for solution component y_i. Set $RTOL_i=10^{-(m+1)}$. Set $ATOL_i$ to the value at which $|y_i|$ is essentially insignificant.

The norm that DASSL uses is a weighted root mean square norm, given by

$$\| v \| = \sqrt{(1/NEQ) \sum_{i=1}^{NEQ} (v_i/wT_i)^2}$$

where

$$WT_i = RTOL_i \, | \, Y_i \, | + ATOL_i$$

When DASSL is finished (either successfully or unsuccessfully), it returns to the user's calling program with a flag IDID that indicates what happened. If the flag is positive, the problem has been solved successfully, otherwise, a negative flag value is produced due to possible errors. The code documentation gives information about the most likely cause of the problem in the event that a negative IDID is encountered.

2.3.3 Algorithm for the Start-up Period

Step 1

The stillpot temperature and stillpot vapour compositions are determined via bubble point calculations using equations (2.45) and (2.44). This fixes the boundary conditions on the variables associated with the vapour phase at the bottom of the column.

Step 2

The entire column is assumed to be filled with vapour having the reboiler vapour compositions, temperature and prescribed flow rate (boilup rate). This fixes the initial values for $y_{i,j}$, T_j and V_j.

Step 3

Using a total condenser fixes the boundary conditions on the variables associated with the liquid phase, at the top of the column. This is based on the composition, flow rate and bubble point temperature of the (condensed) overhead vapour, and also gives initial values to $x_{i,j}$, L_j, $x^*_{i,j}$ and $y^*_{i,j}$ (eqs. (2.60), (2.61), (2.62)).

Step 4

Physical and transport properties such as densities, viscosities, diffusivities, mass transfer coefficients, equilibrium constants and holdups are estimated using the methods given (see Table 2.2).

Step 5

Substituting equations (2.59) and (2.62) into equations (2.60) and (2.61), the variables $x^*_{i,j}$, $y^*_{i,j}$ are eliminated from the differential equations. The ordinary differential equations

(2.60) and (2.61) are reduced to algebraic equation forms using finite difference approximations and written in a matrix form. The size of the matrix is defined by the number of nodes considered in the packed section.

Step 6

The coefficients of the matrix are computed. This matrix is then decomposed using Gauss Elimination method with pivot selection to yield new values for the bulk mole fractions ($x_{i,j}$, $y_{i,j}$) throughout the finite-difference grid. The variables $x^*_{i,j}$, $y^*_{i,j}$ are calculated by the equations (2.59) and (2.62) using the new values of the variables $x_{i,j}$, $y_{i,j}$. If the mole fractions for a particular element (Δz) do not sum to 1.0, they are normalized.

Step 7

Column interfacial temperatures (T_j) are determined via bubble point calculations using eq. (2.45) and the newly computed interfacial mole fractions. The method of false position is used since the boundaries are defined by the boilup temperatures of the pure components that the mixture includes.

Step 8

These new temperatures and compositions are used to compute liquid and vapour specific enthalpies, holdups and physical properties, thus allowing the simultaneous solution of the total energy and material balances, eqs. (2.63) and (2.64), respectively, to yield new total liquid and vapour flow rates (L_j, V_j) throughout the column. The same matrix decomposition method is used as in step 5.

Step 9

The new reboiler holdup and compositions are calculated using eqs. (2.70) and (2.69)

Step 10

Condenser and reboiler heat transfer rates (Q_C and Q_B) are determined by eqs. (2.67) and (2.71).

Step 11

Calculations for the start-up period are continued until the values of the calculated variables are no longer changing from one iterative step to the next, indicating that a steady state operating condition has been achieved. The error is the difference between two successive iterates and 2-vector norms of the errors for each calculated variable should be smaller than a tolerance value (it is taken as 1.0E-6 in this section). At this point, the product period is initiated by making a step chance from a total reflux operation to one with a specified finite reflux ratio.

Step 12

If the steady state condition is not reached, the above sequence of calculations is repeated (from step 6 to 11) using a new set of initial values. They are calculated using a relaxation method based on the new and old values for the dependent model variables. If n_i is a model variable, then the new set of initial values for n_i is $n_i=w*n_i+(1-w)*n_{i,old}$, w is the relaxation factor ($0 \leq w \leq 1$).

2.3.4 Algorithm for the Product Period

When an operation mode is chosen, the reflux ratio is determined. For all operation modes, the simulation of the packed section follows almost the same procedure given below.

Step 1

From a total reflux steady state condition, dynamic operation begins with the first drop of product drawoff. Computationally, this means that the product rate jumps discretely from a value of zero to some finite value, D. For a particular reflux ratio R this value of D would be

$$D = \frac{L_E^o}{R+1}$$

The superscript o denotes steady state total reflux, and the superscript + denotes the instant that product drawoff is begun. With product drawoff the reflux rate immediately drops to $L_E^+ = L_E^o - D$. It can be easily shown that with the assumption of negligible fluid dynamic lags, the flow rates up and down the column become $L_j^+ = L_j^o - D$, $V_j^+ = V_j^o$ and all other initial values are retained from the total reflux calculation. It should be noticed that although the assumption of negligible fluid dynamic lags does not seriously affect the composition profile, it does cause errors in the flow rates calculations, especially in the first several time steps.

Step 2

New values for the bulk and interfacial mole fractions ($x_{i,j}$, $y_{i,j}$, $x^*_{i,j}$, $y^*_{i,j}$) are calculated using eqs. (2.29), (2.30), (2.31) and (2.32) by either matrix decomposition method or DASSL. If the mole fractions for a particular element (Δz) do not sum to 1.0, they are normalized.

Step 3

A new set of interfacial temperatures is determined from eq. (2.45). A corresponding set of interfacial vapour mole fractions is calculated from eq. (2.44).

Step 4

Physical properties, packed section and condenser-reflux drum holdups, and liquid and vapour enthalpies are calculated and holdup and enthalpy derivatives are determined with respect to time, by forward difference approximations.

Step 5

From equations (2.33) and (2.34), a new set of values of liquid and vapour flow rates is computed. At this point, again, either the matrix decomposition method or DASSL can be used.

Step 6

The reboiler molar holdup is computed from eq. (2.58).

Step 7

Steps 2 through 6 are repeated with a corrector step for the same time increment until the 2-vector norms of the errors are equal or smaller than the predetermined error value. This is the inner iterative loop.

Step 8

Condenser and reboiler heat transfer rates are calculated from eqs. (2.38) and (2.42).

Step 9

Steps 2 through 8 are repeated for subsequent time increments until the desired amount of distillate has been withdrawn for all components and a desired stillpot liquid composition has been achieved. This is the outer iterative loop.

2.3.5 Constant Overhead Composition Control Algorithm

The reflux ratio control algorithm is an option of the simulation program PACBACDIS developed in this work. When this option is chosen, the value of reflux ratio is calculated by Hooke-Jeeves direct search method. This value is the one that the overhead composition takes its set point value at the end of the simulation. Hooke-Jeeves method is a direct search method with a univariant approach. It contains two steps: (1) the local exploration step and (2) the acceleration step. The first step is designed to explore the local behaviour of the function. A better search direction is found in this step. The second step is used to accelerate the search along the direction which is found by the local exploration.

The procedure is as follows:

Step 1 A starting point is chosen for the reflux ratio. The first point is also designated as a base point R(0). The initial step size STEP for the local exploration is also chosen.

Step 2 The algorithm given in section 2.3.4 is carried out and the overhead composition is calculated. This step is repeated for every reflux ratio set during this procedure.

Step 3 The error is calculated (ERROR=|overhead composition - set point|). If the ERROR(0) is smaller than its predetermined value, stopping criterion, the reflux ratio is found for this time step. Time is increased. Steps 2 and 3 are repeated. If not, the algorithm goes to Step 4.

Step 4 The search is started for a new reflux ratio by Hooke-Jeeves method. The local exploration is carried out at the initial base point R(0). First, a point R(1), (R(1)=R(0)+STEP), which perturbs R of the base point is tested against the point R(0) to see whether or not ERROR(1) is better than ERROR(0), better means smaller error here. If it is, R(1) is named as a successful point, S. Next the point R(2), (R(2)=R(0)-STEP), is tested by comparing its error value with the error value of the successful point. The better value is kept as a new successful point, S. If neither R(1) nor R(2) is better than R(0), STEP size is decreased by half and Step 4 is repeated.

Step 5 The acceleration step is performed by moving from R(0), the previous base, through successful point S, the current base, to locate a new point by the equation: R(3)=2*S-R(0). The point located by the acceleration step is tested by comparing ERROR(3) with ERROR(S). If it is a success, R(3) becomes the starting point of the next local exploration and the search is repeated from Step 3.

Step 6 If R(3) is worse, S is the starting point and the search is repeated from Step 3.

2.3.6 Notes for the Simulation Package "PACBACDIS"

The simulation package developed, "PACBACDIS", was coded in Fortran77 and the program listing is given in Appendix 11. It is a general purpose simulation package for the simulation and optimization of multicomponent packed batch distillation columns operating under different modes (constant reflux, variable reflux and optimal reflux policy). Besides simulation, PACBACDIS can also accomplish optimization and control of columns. It is flexible and user friendly. PACBACDIS was extensively tested with experimental data involving binary and multicomponent systems with nonideal behaviour and in columns with appreciable holdup effects.

The package allows the user to choose the task (simulation using various models, optimization or control), the operating policy and the model (plug flow or superimposed axial dispersion, with holdup or no holdup).

There are two type of files created or used by PACBACDIS that are of interest to a user.

Input File

The input to PACBACDIS is in the form an input file. The user has to write a Fortran Data File and data entries are read into the program by list-directed READ statements. This means that each data entry is interpreted according to its position in the input file list, so the data should be included in a given order. Physical property data and feed mixture data are entered in a decreasing volatility order. Typical input files written for a binary system (cyclohexane-toluene) "binary.dat" and a ternary system (cyclohexane-n.heptane-toluene) "ternary.dat" are given in Appendix 4. The kind of data input to the system by a user may conveniently be divided into four groups.

Column and Packing Specifications:

These data define the column and packing but not its operation. It includes height of the packing, inner diameter of the column, type of the packing and the packing properties (specific surface area, packing factor, void fraction of dry packing, material and nominal size) and bottom and top column pressures.

Mixture Specifications:

This data defines the mixture charged into the column. It includes the number of components and their names, quantity of initial charge and its composition and the physical property model parameters of each component (coefficients of ideal enthalpy equation, molecular weight, critical temperature, pressure and volume, boiling point temperature, acentric factor, solubility parameter, coefficients of liquid density, liquid viscosity and ideal gas heat capacity equations). The primary source for physical property data is the data bank of Coulson et. al. Vol.6 and Reid et. al. (see Table 2.2 for physical property models used).

Operation Step Specifications:

This data defines the operation of the column during an operation step. It includes specifications such as boilup rate or reboiler heat input rate, condenser and column holdups, operation mode and its parameters (reflux ratio, controller and setpoints).

Simulation Specifications:

This data defines the simulation parameters such as spatial step size, relaxation factor and the coefficients of the parametric study. It also includes a command to indicate whether a total reflux calculation is to be performed as part of the operation step. The names for the output files and the frequency of output data are defined.

Report Files:

They are output files containing details about the progress of the program and all simulation results. The simulation results include complete column profiles (liquid and vapour mol fractions, flow rates, temperatures, pressures, physical and thermodynamic properties and product and reflux specifications) at the end of total reflux calculation and at the end of each operation step at user specified time intervals. The data are recorded into eleven output files whose names are pregiven in the input file by the user. Divisions are made to ease the analysis of the data later. Abstracted sections from report files are given in Appendix 5.

CHAPTER 3

EXPERIMENTAL WORK ON PACKED BATCH DISTILLATION SYSTEM

3.1 Introduction

Most papers published to date on the modelling of batch distillation columns have only presented theoretical predictions and there is still a lack of experimental results for verification (see Table 2.1 in Chapter 2.). Because of the nonavailability of experimental data, many researchers have usually chosen theoretical examples. The trend often used is to vary the input specifications systematically to the simulation and then to evaluate their effects on algorithm predictions qualitatively (e.g. Hitch and Rousseau, 1988). This is regarded as parametric study. Jang (1990) concluded in his thesis that batch distillation experimental data are still needed to verify his simulation program completely, even when the simulation results and the optimization results agree with the theoretical trends.

The objective of this experimental study is to produce data for the complete verification of the model (simulation procedure is given in Chapter 2) and to study different operation modes for a batch distillation column. Two operation modes were studied and these are constant reflux ratio and constant overhead composition. Experiments performed were intended to cover three main areas. The first group of experiments was carried out to test the effect of reboiler holdup, charge mixture initial compositions and different constant reflux ratio operation modes on the separation. The second group was used for designing the overhead composition control for the column and the third one actually tested how two different operation modes, constant reflux and constant overhead composition, improve the separation of a given mixture.

Two well-known test mixtures were selected for experimental work. A binary mixture (Cyclohexane-Toluene) was used for developing a control program and right operation conditions of the experimental procedure. A ternary mixture (Cyclohexane-n.Heptane-Toluene) was selected to test multipurpose and multicomponent aspects of the model. Their physical properties and operating conditions were available.

A pilot plant scale batch distillation column was extensively modified before the present experiments were performed. The experimental system had the following shortcomings before the modifications:

The column was packed with 9 mm ceramic Raschig rings that are less efficient in types of random packings.

The original reflux system solenoid valve was actuated by a timer by changing the open & shut cycle times of the valve. The timer could be operated only manually and would not allow continuous control of the reflux ratio.

There was no liquid distributor in the system leading to maldistribution of liquid in the packed section.

The column packings were badly corroded and gave poor performance.

Only five available thermocouples were placed in the reboiler, top of the column, reflux line and cooling water inlet and outlet of the system. These would not supply enough data for the system with the required accuracy. Their readings could be obtained from a digital thermometer only manually.

Sampling points from the reboiler and reflux line could be operated only manually because of the existing ball valves. A single operator could not take samples from both lines at the same time at required intervals.

These shortcomings and deficiencies were overcome by extensive modifications and the rig was automated by a microcomputer and a data control and acquisition system with great emphasis on a safe design. It was decided that the system would also be used by novice undergraduate students for their practical laboratory work and therefore safety was the highest priority.

The improved experimental system and the control software written for this work are given in the next sections. Also given are the instructions to run the software and the experiments performed.

3.2 Description of The Experimental Apparatus

The schematic diagram of the experimental apparatus for the study of packed batch distillation operation modes and control is illustrated in Figure 3.1. Each item in the diagram with its capacity, operation range and usage is explained as follows:

R1: Flow meter on the cooling water inlet to the condenser with [0.1-1.4liter/min] range. It was installed in parallel with the existing one R2 to facilitate efficient manual control.

R2: Flow meter on the cooling water inlet with [0.5-5 liter/min] range.

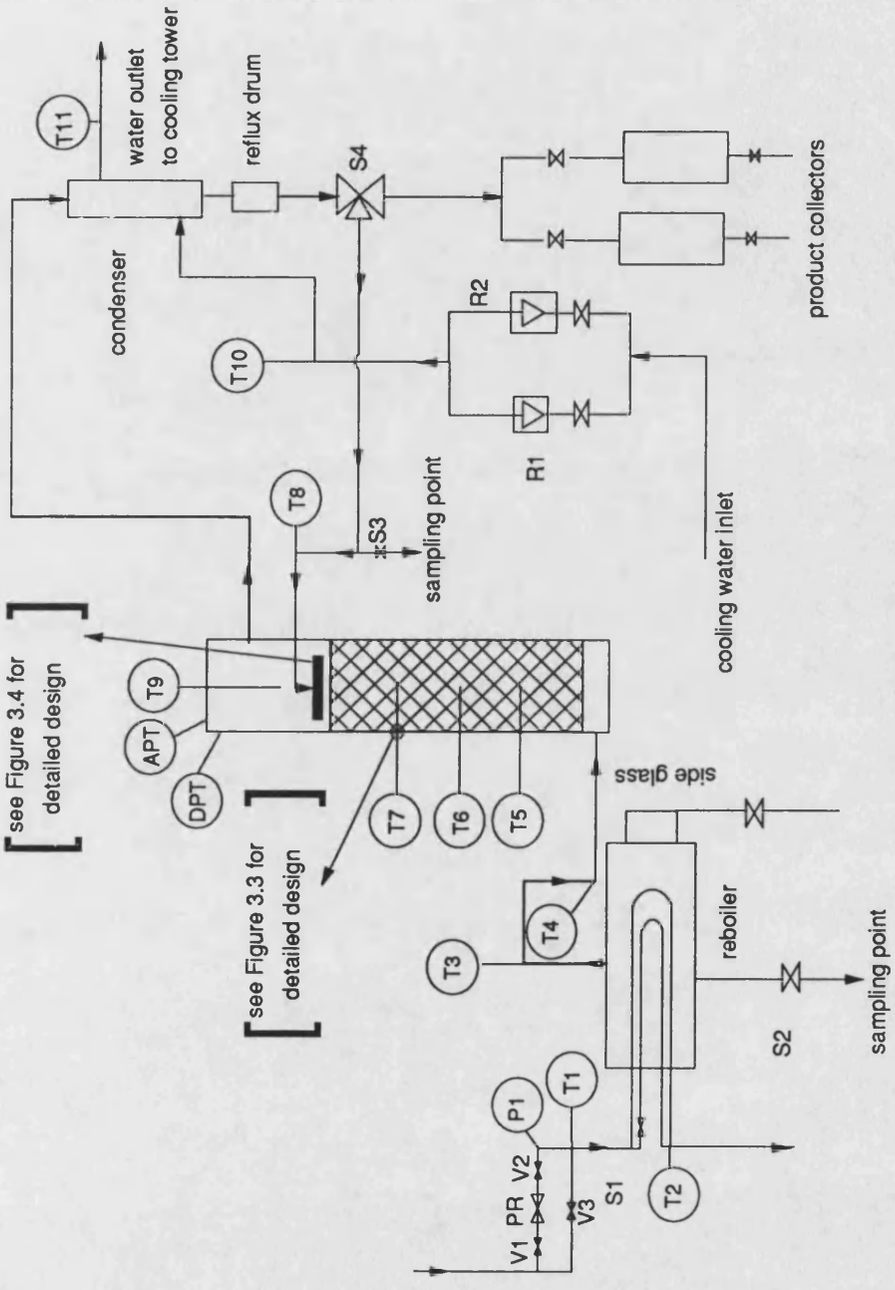


Figure 3.1 Schematic diagram of experimental packed batch distillation system [refer next pages for details of symbol specifications].

Column: Figure 3.2 shows the schematic diagram of the detailed packed column section of the experimental system. The column is 2.48 meter high and 8 centimetres in diameter. It is made of mild steel and the installed packing height is 1.7 meter.

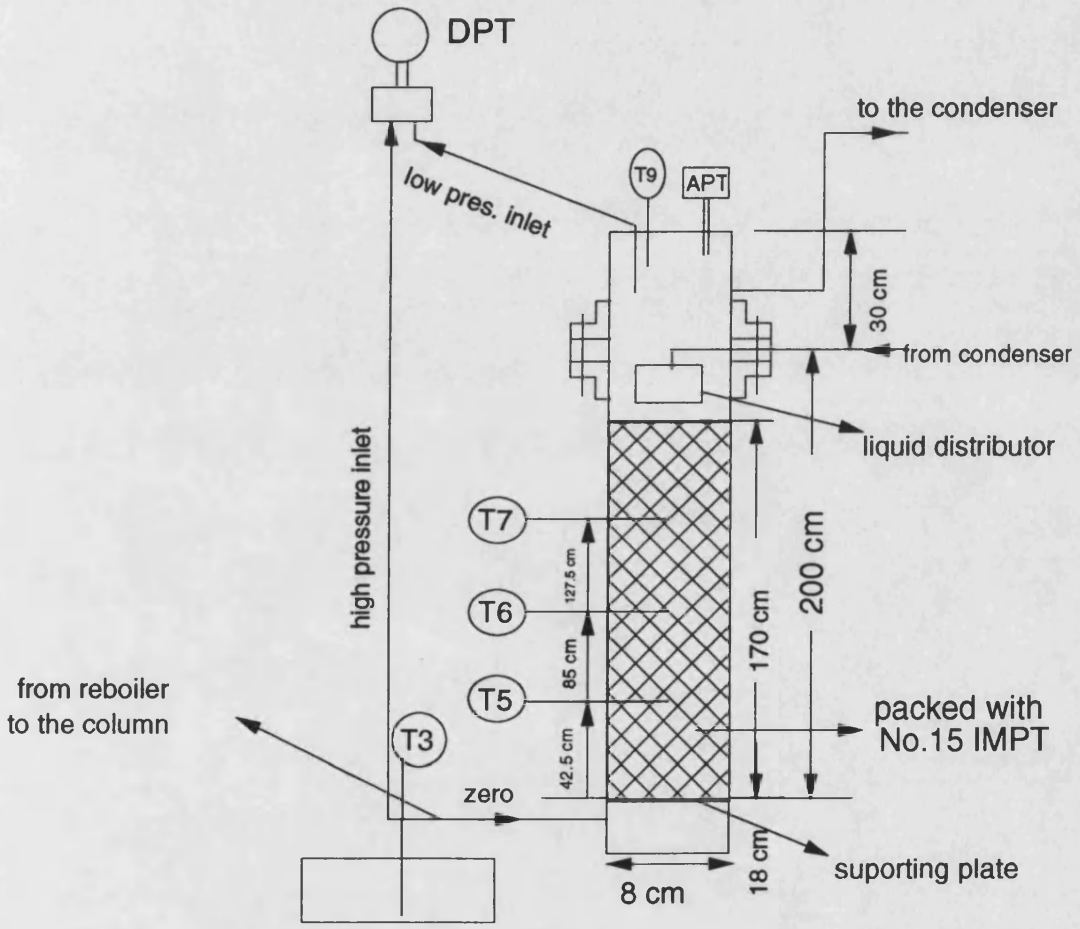


Figure 3.2 Schematic diagram of detailed packed column section.

Reboiler: Mild steel, 30 litre capacity and equipped with two "U" shaped steel pipes with 0.170 m^2 total heat transfer surface area (0.83 inch outer diameter and length 170 cm and 88 cm). A boiler located in the laboratory supplies saturated steam at a maximum pressure of 30 lb/in^2 g (44.7 psia or 3.5 bar) to the reboiler.

Condenser: A shell and tube heat exchanger with condensing vapour in tubes and cooling water in the shell. The cooling water is supplied by a pump that draws water from a cooling tower. Used condenser water is recycled back to the system.

PR: A Pressure Regulator was used for adjusting the steam flow rate to the reboiler manually.

V1,V2,V3: Ball Valves on the Steam lines to the Reboiler.

P1: Pressure Gauge on the steam inlet line

S1: One-way, normally close solenoid valve installed on the 170 cm and 0.83 inch steam pipe to the reboiler for adjustment of heating area.

S2: One-way, normally close solenoid valve on the sampling point from the reboiler.

DPT: Differential Pressure Transducer and Transmitter with standard accuracy ± 0.1% span, installed between the reboiler and top of the column (see Figure 3.2), correlated for [0-20 mm Water] span. It is used for measuring the pressure drop across the column to supply data for the control of boilup rate.

APT: Absolute Pressure Transmitter with standard accuracy ± 0.2% span, correlated for [0-2 bar] span. It is used for the top column pressure measurements.

S3: One-way, normally close solenoid valve on the sampling point from the reflux line.

S4: 3-way solenoid valve on the reflux line and it is operated as a reflux splitter. It normally diverts all the condenser material to the column as reflux. When it is energized, it diverts all the condenser material to the product collectors.

T1-T11: The experimental system was equipped with 11 thermocouples and their locations are shown in Figures 3.1 and 3.2 and the specifications are listed in Table 3.1. Extra care had to be taken when handling column section installations and the detailed design is illustrated in Figure 3.3. The thermocouples were calibrated prior to use and the calibration results are given in Appendix 6. The corrected temperature readings are provided by the control program "DISTCONT".

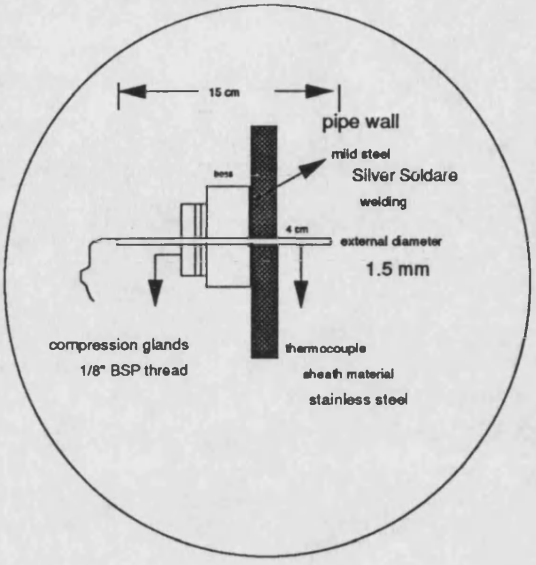


Figure 3.3 Detailed design of a thermocouple location in column section.

Liquid Distributor: A liquid distributor was designed so that a uniform liquid distribution could be achieved over the tower packing for developing the full efficiency of the packing (a detailed diagram is given in Figure 3.4).

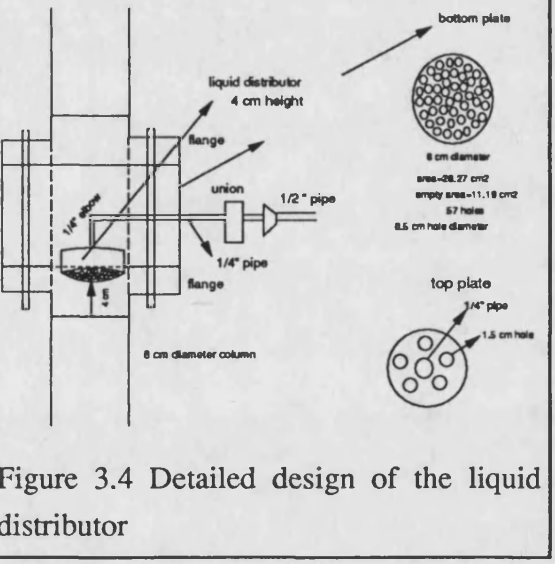


Figure 3.4 Detailed design of the liquid distributor

Table 3.1 Thermocouple Specifications and Locations for the Modified System

Thermocouple Material	:	Nickel Chromium / Nickel Aluminium Type K
Maximum Operating Temperature	:	800 C
Sheath Material	:	Stainless Steel 25/20 Chromium Nickel, Steel to BS 970 Grade 310 524

Bonded Junction

1.5 mm External Diameter

Tail Length 100 mm

A (route length cm) values are given below for each thermocouple.

11 T/C Compression Gland TGB 151 (SS Pressure Entry Glands 1/8 BSP)

T1	Steam inlet to the reboiler (A:15 cm)
T2	Condensate outlet from the reboiler (A:15 cm)
T3	In the reboiler liquid phase (A:50 cm)
T4	Vapour inlet to the bottom of the column from the reboiler (A:15 cm)
T5	In the column, 42.5 cm height from the packing supporting plate (A:15 cm)
T6	In the column, 85 cm height from the packing supporting plate (A:15 cm)
T7	In the column, 127.5 cm height from the packing supporting plate (A:15 cm)
T8	Reflux line to the column from the condenser (A:15 cm)
T9	In the column, top of the packing (A:50 cm)
T10	Cooling water inlet to the condenser (A:15 cm)
T11	Cooling water outlet from the condenser (A:15 cm)

Packing: The packing was changed from ceramic Raschig rings to No.15 Intalox Metal Tower Packing (IMTP) (supplied by Norton Company), taking advantage of its lower pressure drop and greater maximum capacity than equivalent sizes of the other packings to improve the separation efficiency. The use of IMTP reduces the overall capital and operating costs, as the column will be able to achieve the same throughput for less energy input or greater throughput for the same energy input. No.15 Intalox Metal Tower Packing (15 mm) is the recommended size for the 8 cm-diameter column (Column diameter < 0.3m then use packing size < 25 mm).

Microcomputer: A BBC microcomputer with an Analog to Digital Converter was used for running the control program and data acquisition.

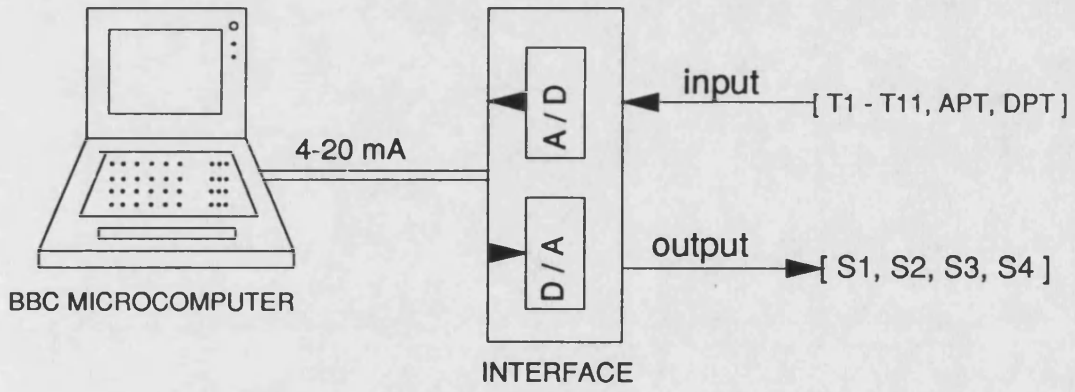


Figure 3.5 Connection diagram between the microcomputer and the system

Interface: It is a Field 1000 series RS485 version that is a flexible and compact interface system. The standard unit is a dual housing unit that accommodates up to three eight-channel input or output boards, giving a capacity of 24 channels. The boards may be any mix of analogue input, digital input and digital output boards. The interface accommodates 8 channel I/O boards: A Analogue amplifier and master unit board, B

Analogue slave A input board, J Digital output board (Reed relay switching 250 mA DC) and a RS232C/RS485 passive converter. The microcomputer, interface and system (the thermocouples (T1-T11), the pressure transducers (DPT and APT) and the solenoid valves (S1-S4)) connections and the input and output signals are summarized in Figure 3.5. All the wiring from the column passes through conduit to a wall panel box.

3.3 Computer Control Program "DISTCONT" for Distillation Experiments

Appendix 7 shows a listing of the computer program DISTCONT used for controlling the packed batch distillation experiments. DISTCONT was written in BBC basic and permits startup of the column, transition to PI control and data acquisition. During runtime, the program displays all process data, available control options and their current values as shown in Table 3.2 and they are updated every five seconds on the screen. A hard copy can be taken from the printer on the line or written on a disc at time intervals ranging from five seconds upwards.

In DISTCONT, the values of operation parameters could be entered or modified by the operator at any time during the run through an interrupt system. A letter is used for each parameter and entering any of these letters from the keyboard followed by the value with no space between will change the old settings. Detailed information for the setting letters is given in Table 3.3.

Table 3.2 Example Control Screen for the Control Program DISTCONT.

TEMPERATURES [C]	MT	CT	Elapsed time	6770	R-set reflux ratio
STEAM_INLET	122.3	122.8	Filename	JULY3	S-set sampling time
CONDENSATE_OUT	122.9	123.9	Save every	15	O-open disc file
MIXLIQ_REBOILER	93.9	94.6	Sample each	300	C-close disc file
BOILUP_REBOILER	94.8	95.6	Writing each	300	P-pressure set point
PACKING(42.5CM)	85.7	85.7	Setpointpres	12	H-heat on/off
PACKING(85CM)	83.2	83.2	Setpointxtop	0.90	L-control loop on/off
PACKING(127.5CM)	81.1	81.1	Sample time	0.4	D-disc write period
LIQ_REFLUX	44.5	46.2	Reflux	70	T-take top sample
TOP_COLUMN	77.7	78.6	Steam	ON	B-take bottom sample
COOL_WATER_OUT	23.4	25.1	Control heat	ON	G-take both samples
COOL_WATER_IN	9.3	11.1	Control conc	ON	F-set sample period
			Gain	264	W-set print period
			Time Constant	146	I-Comp.Cont.Loop on/off
			Comp.	1	E-Conc.set point
					K-Set proportional gain
Diffpress	10.92	[mm Water]			M-Set integral time cons.
Abspress	0.986	[bar]			N-Set comp. (C:1/H:2/T:3)
More Vol. Comp. Mol Frac.(Top Tem.)=0.9917					

MT-Measured Temperatures

CT-Corrected Temperatures

Time is in seconds.

Table 3.3 Operation Parameters of DISTCONT

R sets the reflux ratio in constant reflux ratio operation mode. The value entered after 'R' is equal to L/V %. Default value for R is 100 that indicates a total reflux condition. This means that S4 (three-way reflux line solenoid valve) diverts all the liquid flow from the condenser to the column as reflux ('OFF'). When R is set to 0, S4 is energized to divert all the flow to the product collectors ('ON'). Any value between 0 and 100 implies the 'OFF'-'ON' cycle times of the reflux valve in ten second time periods. The ratio of these two time periods is the reflux ratio (i.e. 'R70'means that S4 is 'OFF' for 7 seconds and 'ON' for 3 seconds in every 10 seconds. The reflux ratio is L/D=7/3=2.33.). The value typed after 'R' must be an integer. The current value of 'R' is monitored as 'Reflux' on the screen.

S sets the length of time in seconds for liquid sampling. The solenoid valves S1 and S3 will be kept actuated in this length of time. Its default value is 0.4 and is monitored as 'Sample time'.

O opens a filename. Acquired data may be saved on a floppy disc under this name. The filename must be specified after typing 'O'. It is possible to use more than one file in a session for output purposes. However the current file must be closed before a new one is opened. The current filename is monitored as 'Filename' on the screen.

C closes the current data acquisition file. Only 'C' is required, no file name is necessary.

P sets the value of desired differential pressure drop across the column for heat control loop. It is required only when 'Control heat' is 'ON'. The unit for P is in mm Water. The current value of 'P' is shown as 'Setpointpres' on the screen.

H shifts the condition of the solenoid valve S2 on the steam line that is normally close. It could be actuated to open by entering 'H1' for 'ON' or to close by 'H0' for 'OFF'. Unless the 'Control heat' is 'ON', its condition remains as it is. Default value is 'OFF'. The current situation is given as 'Steam' on the screen.

L switches the heat control loop 'ON' by entering 'L1'. The control loop is for keeping the boilup rate constant. The solenoid valve S2 is actuated automatically to keep the pressure drop across the column at its predefined value by 'P'. Default value for L is 0 and could be seen on the screen as 'OFF' for 'Control heat'.

D sets the time interval in seconds for data recording on disc under a predefined filename. Its current value is displayed on the screen as 'Save every'.

T If required, a liquid sample could be taken from reflux line at any time. Entering 'T' will actuate the solenoid valve S3 and keep it on for a predefined length of time.

B If required, a liquid sample could be taken from the reboiler at any time. Entering 'B' will actuate the solenoid valve S1 and keep it on for a predefined length of time.

G If required, a liquid sample could be taken from both the reboiler and the reflux line at any time. Entering 'G' will actuate the solenoid valves S1 and S3 and keep them on for a predefined length of time.

F sets the time interval in seconds for automatic liquid sampling from the reboiler and the reflux line. Both solenoid valves S1 and S3 are actuated for sampling at each time interval. Default value is 0 that implies that no automatic sampling is required. The current value of 'F' is monitored as 'Sample each' on the screen.

W sets the time interval in seconds for data recording as hard copy by a printer on the line. Its current value is shown on the screen as 'Writing each'. The monitored data is recorded on a single line as follows: Elapsed time in seconds, corrected values of T1,T2,T3,T4,T5,T6,T7,T8,T9,T11,T10 thermocouple readings, differential pressure drop in mm water, absolute top column pressure in bar, predicted value of more volatile component overhead composition. The value of reflux ratio is written at the end of the line only if it is different from the last one.

I switches the operation mode from constant reflux ratio to constant overhead composition control. 'I1' sets PI control loop 'ON' to keep the predicted overhead composition at its predetermined value. Entering 'I0' returns the operation mode to constant reflux ratio. The current operation mode is shown as 'Control conc' on the screen.

E sets the overhead composition. The value must be between 0 and 1. Its current value is shown as 'Setpointxtop' on the screen.

K sets the proportional gain of PI controller. Its current value is shown as 'Gain' on the screen.

M sets the integral time constant of PI controller. Its current value is shown as 'Time Constant' on the screen.

N sets the component that is currently taken as a product. Predefined value of 'Setpointxtop' corresponds this chosen component. Options are as follows: 1 for Cyclohexane - 2 for n.Heptane - 3 for Toluene. Its current value is shown as 'Comp.' On the screen.

3.4 Experimental Procedure

The startup procedure is outlined as follows:

1. The reboiler is charged with the test mixture that composition and amount are known.

2. Switch on the main electricity and bring the BBC Microcomputer on line. Run the control program 'DISTCONT', choose the operation mode and enter required data. A typical screen layout can be seen in Table 3.2. Program variables are listed in Table 3.3.

3. Turn on the cooling water to the condenser by opening the valve on the line manually. Depending on the required flow rate of the water, use R1 or R2 flow meter line.

4. Admit steam to both heating coils in the reboiler by opening manually the valves V1 and V2 on the steam inlet (V3 is close) and actuate solenoid valve S1 by entering 'H1' from the computer keyboard. This entry is given as 'H-heat on/off' in Table 3.2. Select a steam pressure by adjusting PR manually, available maximum steam pressure is 30 psig (read the steam pressure by using P1 pressure gauge) (see Figure 3.1 for valve locations).

5. Observe the temperature of the liquid in the reboiler (T3, MIXLIQ-REBOILER value on the screen) and of the vapour from the reboiler (T4, BOILUP-REBOILER value on the screen) (see Table 3.2).

6. When the temperatures T3 and T4 reach the mixture boiling point temperature, observe the temperatures T5 (PACKING(42.5 cm)), T6 (PACKING(85 cm)), T7 (PACKING(127.5 cm)) and T9 (TOP-COLUMN) through the column and the DPT reading (Diffpress [mm Water]) of differential pressure value to follow the progress of vapour in the column.

7. When the vapour reaches the condenser (it could be observed from the raised temperature difference between T11 (COOLING-WATER-OUT) and T10 (COOLING-WATER-IN)), adjust the pressure regulator PR to keep the steam pressure (P1) at a value that keeps the Diffpress reading constant. This Diffpress value is selected to obtain the minimum holdup in the reflux accumulator (observe the QVF glass reflux container under the condenser, see Figure 3.1).

8. Adjust the flow of cooling water through the condenser by using R1 or R2 so as to obtain a suitable temperature rise (observe T11 (COOLING-WATER-OUT) and T10 (COOLING-WATER-IN) and set a temperature rise between 5-10 C).

9. Run the column at total reflux conditions (default value of Reflux=100 indicates the total reflux operation, see Table 3.3). Meanwhile check temperature profile, pressure readings and take samples from sampling points by actuating S2 and S3 solenoid valves (see Figure 3.1). Analyse the samples by an appropriate technique.

10. When a steady state temperature profile and the specified overhead composition is reached, the column is ready to run at a selected operation mode. Ensure that product collectors are ready and appropriate valves are open.

The product withdrawal procedure is outlined as follows:

1. Select the operation mode (constant reflux ratio or constant overhead composition) by entering required data.

   Constant Reflux Ratio Operation Mode: 'Control conc' must be monitored as 'OFF' on the screen, if not, enter 'I0'. Set the value of reflux ratio by 'R'.

   Constant Overhead Composition Operation Mode: Determine overhead composition by 'E' and proportional gain by 'K' and integral time constant by 'M' and then switch the PI controller by 'I1'.

2. Observe Diffpress value and adjust PR manually to keep it constant at a specified value. Check the temperature difference across the condenser and adjust R1 or R2 accordingly.

3. Change the product collectors accordingly to collect main products and slop cuts separately.

4. Temperature and pressure readings, and samples from S2 and S3 can be taken automatically with given intervals and can be recorded as a hard copy using a printer on the line or saved into a file.

5. Check the side glass of the reboiler at regular intervals and do not let reboiler run dry.

Shut-down procedure:

1. Close steam valve V1, leave the cooling water running until T3 MIXLIQ-REBOILER reading reaches a value less than 30 C.

2. Close the cooling water valve. Switch off the main electricity.

3. Drain the reboiler, column and product collectors into the liquid container provided after completed required measurements.

3.5 Sample Analysis Equipment and Procedures

Before composition can be successfully controlled, it must be measured. This section deals with two methods used for analysing binary and ternary test mixtures.

3.5.1. Refractometer for Analysis of Binary System

An ABBE 60 Refractometer was used for the analysis of the binary text mixture. The instrument employs the critical angle effect marked by a demarcation line, generally known as the borderline, between light and dark portions of the telescope field. The rays constituting the critical angle, on emerging from the prism, fall on a mirror where they are reflected into the field telescope of the instrument. The position of the mirror required to bring the borderline into coincidence with the telescope cross-wires is indicated by a moving scale observed in the scale telescope. This scale is the direct function of refractive index and in the standard model is divided in such terms. Since the scale is non-linear no means of sub-division can be employed and therefore the degree scale model carries an evenly divided scale capable of accurate sub-division, enabling refractive index values of high accuracy to be obtained from tables.

The polished surface of the fixed prism is carefully cleaned to prepare it for the sample. A small drop of sample is placed in the centre of the polished surface that should now be placed polished side down onto the surface of the fixed prism. The liquid will spread out to cover the interface. With the eye at the field telescope, the control knob is turned to a position where the observed field is divided into light and dark portions, the dark area below. The borderline is set accurately on the intersection of the cross-wires, sharply in focus by adjustment of the eyepiece that is of a draw tube type. Having obtained this condition, the scale reading is observed in the scale telescope. All readings are relative to Mean Sodium D lines 5893 A. Refractive Index measurement is then obtained by reference to the appropriate calibration table for the spectral line concerned. After taking a measurement, the prism box should be opened and the sample should be cleaned off immediately from both halves.

A calibration curve for the refractive index via Cyclohexane concentration of the mixture was developed from the prepared test samples and given in Appendix 8.

3.5.2 Analysis of Ternary System by Gas Chromatograph

A Pye Unicam Series 204 gas chromatograph (GC) with a flame ionisation detector (FID) was used for analysis of the ternary test mixture. A non-polar packed column (2.8 meter 20% on chromosorb WHP 80-100#) gave an adequate separation of the test mixture components (Cyclohexane-n.Heptane-Toluene).

The signal from the FID of the GC was displayed on a chart recorder. A digital integrator was used for integrating the area between the signal and the base line. The operating conditions are stated in Table 3.4 Calibration results and GC plotter output samples are given in Appendix 9.

Table 3.4 Operating Conditions for the GC Used for Analysis of the Ternary Mixture

Carrier Gas: Helium at 24 psig (30 ml/min)
Detector Gases:
Hydrogen: 11 psig (62.07 ml/min)
Air: 60 psig (317.6 ml/min)
Temperatures
Column: 90 C
Temperature Controller:
Initial time: 3 minutes keep at 90 C
Rate: 32 C/min
Final: 180 C
Final time: 1 min keep at 180 C
Detector temperature: 200 C
Injector Temperature: 200 C
Settings on Ionisation Amplifier
ATT: 128 10^3 ±depress
sample amount: 0.5µlitre
internal standard component: o-Xylene

3.6 Experimental Data

Experiments were conducted using a binary and a ternary mixtures with different operating conditions, charge specifications and operation modes. The experimental data are given in Table 3.5 for the binary mixture and in Table 3.6 for the ternary mixture. It should be noted that 1% reflux ratio (=L/V) was used for most of the experiments during cut-point period to shorten the bottom product purification time. This operating policy was suggested by Wang (1986) as zero reflux ratio during cut-point period. Reflux line instrumentation did not allow to set a zero reflux ratio, therefore the smallest value 1% was used.

Table 3.5 Operating Conditions and Experimental Results of the Binary System

Exp. No.	Feed B_o, mol	Feed Comp. X_{Bo}, mol%	Operation Mode	Data Files	Elapsed Time min	R% L/V	D kg *mol*	X_D mol %	B kg *mol*	X_B mol %	S kg *mol*	X_S mol %	H kg *mol*	X_H mol %
RUN-1	107.13	61.18	Constant Reflux	AP25	27.5 17.5	50 1	5.38 *62.84*	81.83	 1.06 *11.66*	 15.07	1.81 *20.56*	51.4	 0.62 *6.79*	 10.32
RUN-2	115.6	46.37	Variable Reflux	AP26	3.75 4.25 6.67 11.58 7.92 13.5 3.75 9.83	40 50 60 70 80 90 95 1	 5.56 *65.14*	 85	 1.75 *19.15*	 8.2	 0.87 *9.785*	 40.4	 0.66 *7.20*	 6.13
RUN-3	97.8	49	Constant Reflux	MAY1	27.58 18.67	70 1	3.50 *41.15*	88.7	 1.54 *16.86*	 9.62	2.60 *29.27*	41.26	 0.64 *6.99*	 6.45
RUN-4	50	52	Constant Reflux	MAY2	11.08 15.0	70 1	1.34 *15.76*	88.99	 1.13 *12.42*	 13.95	2.59 *29.45*	52.33	 0.65 *7.15*	 14.99

Table 3.5 continued

Table 3.5 continued

Exp. No.	Feed B_o, mol	Feed Comp. X_{Bo}, mol%	Operation Mode	Data Files	Elapsed Time min	R% L/V	D kg *mol*	X_D mol %	B kg *mol*	X_B mol %	S kg *mol*	X_S mol %	H kg *mol*	X_H mol %
RUN-5	136.3	51.51	Constant Reflux	MAY3	24.83 24.17	70 1	3.15 *36.98*	87.14	 4.03 *44.23*	 12.74	5.11 *57.70*	44.79	 0.65 *7.16*	 17.12
RUN-6	100	38.09	Variable Reflux	MAY9	3.667 7.667 9.0 12.5 13.25	50 70 80 90 1	0.53 0.92 0.79 0.51 2.75 *32.06*	92.72 75.62 76.5 78.15 79.6	 3.67 *40.12*	 8.29	1.54 *17.54*	33.08	 0.64 *6.999*	 8.58
RUN-7	100	30.22	Constant Reflux	MAY10	39.833 7.75	90 1	1.65 *19.45*	91.44	 5.46 *59.91*	 12.39	0.9 *10.1*	37.5	 0.64 *7.01*	 9.92
RUN-8	100	32.99	Constant Reflux	MAY15	8.917 27.333	70 1	1.10 *12.91*	87.2	 2.34 *25.83*	 19.17	4.6 *51.56*	36.58	 0.63 *6.9*	 10.46

Table 3.5 continued

Table 3.5 continued

Exp. No.	Feed B_o, mol	Feed Comp. X_{Bo}, mol%	Operation Mode	Data Files	Elapsed Time min	R% L/V	D kg *mol*	X_D mol %	B kg *mol*	X_B mol %	S kg *mol*	X_S mol %	H kg *mol*	X_H mol %
RUN-9	100	50	Constant Reflux	MAY16	6.17	50	1.16 *13.63*	87.8			5.31 *60.18*	48.77		
					29.83	1			1.30 *14.25*	11.48			0.65 *7.14*	13.73
RUN-10	100	49.66	Constant Reflux	MAY17	67.17	90	2.93 *34.67*	95.59			1.62 *18.09*	32.12		
					14.58	1			3.33 *36.43*	9.03			0.64 *7.00*	9.36
RUN-11	100	52.45	Variable Reflux	MAY20	3.0	50	0.52	93.68			0.39 *4.504*	69.53		
					4.4167	60	0.72	87.44						
					10.75	70	1.42	87.14						
					11.58	80	1.01	88.11						
					16.25	90	0.66	86.63						
					46.083		4.33 *50.88*	88.12						
					17.42	95			3.21 *35.1*	8.56			0.64 *7.00*	8.80

Table 3.5 continued

Table 3.5 continued

Exp. No.	Feed B_o, mol	Feed Comp. X_{Bo}, mol%	Operation Mode	Data Files	Elapsed Time min	R% L/V	D kg *mol*	X_D mol %	B kg *mol*	X_B mol %	S kg *mol*	X_S mol %	H kg *mol*	X_H mol %
RUN-12	100	50	Step Changes	JULY2-1 JULY2-2 JULY2-3	10 10 10 10 10 10 10	70 75 80 85 90 95 100		90		18.97				
RUN-13	100	50	Control SP=92%	JULY4A-1 JULY4A-2	38.58 5.667		4.04 *47.64*	92.03	3.62 *39.68*	11.32	0.51 *5.8*	52.92	0.64	
RUN-14	100	50	Control SP=95%	JULY4B-1 JULY4B-2 JULY4B-3	47.75 9.75		3.27 *38.71*	96.2	2.47 *27.04*	10.09	0.94 *10.65*	48.37	0.64	

Table 3.6 Operating Conditions and Experimental Results of the Ternary System

Exp. No.	B_o, mol	X_{Bo}, mol%	Operat. Mode	R% L/V	Data Files	Sample No.	Elapsed Time min	XDCyc mol %	XDHep mol %	XDTol mol %	XBCyc mol %	XBHep mol %	XBTol mol %
RUN-15	98.26	36.742	Constant	70	JUNE4-1	1	4.083	70.309	9.039	20.652			
		28.428	Reflux		JUNE4-2	2	9.083	75.525	10.676	13.799			
		34.830				3	14.083	72.511	14.648	12.842			
						4	19.083	67.214	19.204	13.582			
						5	24.083	62.699	22.565	14.736			
						6	29.083	58.220	25.426	16.353			
						7	34.083	52.770	28.845	18.385			
						8	39.083	45.969	32.600	21.431			
						9	44.083	36.388	36.605	27.008			
						10	49.083	26.62	39.205	34.176			
						11	54.083	17.454	37.183	45.364			
						12	59.083	12.908	32.256	54.837			
						13	65.500				8.479	12.682	78.838

Table 3.6 continued

Table 3.6 continued

Exp. No.	B_o, mol	X_{Bo}, mol%	Operat. Mode	R% L/V	Data Files	Sample No.	Elapsed Time min	XDCyc mol %	XDHep mol %	XDTol mol %	XBCyc mol %	XBHep mol %	XBTol mol %
RUN-16	99.045	44.7 23.5 31.8	Constant Reflux	90	JUNE6-1 JUNE6-2	1 2 3 4 5 6 7 8 9 10 11	49.833 59.833 69.833 79.833 89.833 99.833 109.833 119.833 129.833 139.833 148.25	86.443 85.686 83.72 79.514 73.349 64.771 53.219 40.01 29.135 17.534	6.145 8.046 9.873 13.565 18.438 25.492 35.021 45.33 52.988 58.83	7.411 6.268 6.404 6.92 8.213 9.737 11.76 14.66 17.878 23.63	5.528	8.787	85.685

Table 3.6 continued

Table 3.6 continued

Exp. No.	B_o, mol	X_{Bo}, mol%	Operat. Mode	R% L/V	Data Files	Sample No.	Elapsed Time min	XDCyc mol %	XDHep mol %	XDTol mol %	XBCyc mol %	XBHep mol %	XBTol mol %
RUN-17	98.18	46.112	Constant	92	JUN13-1	1	1.417	93.35	2.56	4.09	26.66	24.66	48.67
		22.218	Reflux		JUN13-2	2	11.333	93.86	2.32	3.82	29.9	25.07	45.02
		31.663			JUN13-3	3	21.333	93.719	2.39	3.89	28.199	22.85	48.95
						4	31.333	92.41	3.44	4.15	30.398	23.767	45.83
						5	41.333	91.225	4.405	4.37	31.99	22.13	45.88
						6	51.333	90.62	4.95	4.43	30.94	22.17	46.88
						7	61.333	89.7	5.64	4.67	30.80	21.35	47.85
						8	71.333	87.94	7.03	5.03	31.45	19.87	48.68
						9	81.333	86.32	8.31	5.37	28.53	19.72	51.75
						10	91.333	82.93	10.68	6.39	22.72	19.81	57.47
						11	101.333	79.27	13.96	6.77	26.24	17.46	56.31
						12	111.333	74.63	17.74	7.64	21.89	16.85	61.26
						13	121.333	68.42	22.87	8.72	18.43	15.48	66.1
						14	131.333	56.84	32.51	10.66	14.34	14.52	71.14
						15	141.333	42.53	43.82	13.63	9.95	12.98	77.07
						16	151.333	28.49	54.8	16.72	8.43	11.61	79.96
						17	161.333	19.38	60.8	19.81			
						18	171.333	13.36	62.73	23.91	6.98	11.9	81.13
						19	181.333	8.43	59.45	32.12	5.27	6.506	88.22

Table 3.6 continued

Table 3.6 continued

Exp. No.	B_o, mol	X_{Bo}, mol%	Operat. Mode	R% L/V	Data Files	Sample No.	Elapsed Time min	XDCyc mol %	XDHep mol %	XDTol mol %	XBCyc mol %	XBHep mol %	XBTol mol %
RUN-18	99.83	43.849 21.230 34.930	Constant Reflux	95	JUN20-1 JUN20-2 JUN20-3	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18	12.333 27.333 42.333 57.333 72.333 87.333 102.333 117.333 132.333 147.333 162.333 177.333 192.333 207.333 222.333 237.333 252.333 267.333	68.84 81.23 87.35 90.07 90.31 90.52 89.13 88.12 86.55 82.12 79.55 71.97 60.34 64.18 26.82 16.31 8.78 6.51	10.63 6.76 4.66 4.25 4.83 4.91 6.02 6.78 8.04 11.76 13.81 20.14 30.00 43.14 56.94 63.61 66.31 61.49	20.53 12.01 7.99 5.68 4.86 4.57 4.85 5.095 5.4 6.12 6.64 7.89 9.66 12.68 16.24 20.08 24.91 32.00	37.27 37.41 36.51 37.16 34.47 32.07 29.05 24.52 24.47 20.91 17.96 15.33 13.43 10.71 12.10 7.99 8.58 8.44	21.13 20.66 21.71 20.98 23.39 23.97 24.22 25.44 25.48 26.49 26.87 26.00 25.06 22.69 18.16 18.34 17.66 12.16	41.6 41.92 41.79 41.87 42.14 43.95 46.73 50.04 50.05 52.6 55.17 58.67 61.51 66.61 69.73 73.67 73.76 79.4
RUN-19	97.8	39.50 27.49 33.01	Control SP-Cyc= 92%		JUL10-1 JUL10-2 JUL10-3		54.92						

3.7 Control System

This section discusses the design of sampled data controller for the experimental batch column. The system identification procedure and the tuning of PI controller parameters for the overhead composition control are explained. The objective of this system has to produce experimental data for the constant overhead operation mode and therefore the control problem was not studied extensively. The binary system (Cyclohexane-Toluene) was used for the control experiments. Figure 3.6 shows the experimental column with control configurations. Ki (1986) and Bozenhardt (1986 and 1987) were followed while selecting control variables and control loops.

Sampled-data control defines control action that is executed at discrete time intervals. The controller output changes each time it receives a measurement and then holds that value until a new measurement is received. Digital computers are ideally suited to this type of control because they operate at discrete intervals.

The control objective is to operate the experimental column to yield constant overhead composition (OC) at given setpoint. The reflux ratio (R) must be increased through the cycle to obtain constant product concentration as the reboiler mixture concentration decreases.

In a digital computer-control system, what we need is an equation or algorithm that can be programmed into the digital computer. At the sampling time for a given loop, the computer looks at the current process output OC(t), compares it to a setpoint, and calculates a current value of the error e(t). This error, plus some old values of the error and old values of the controller output or manipulated variable that have been stored in computer memory, are then used to calculate a new value of the controller output R(t).

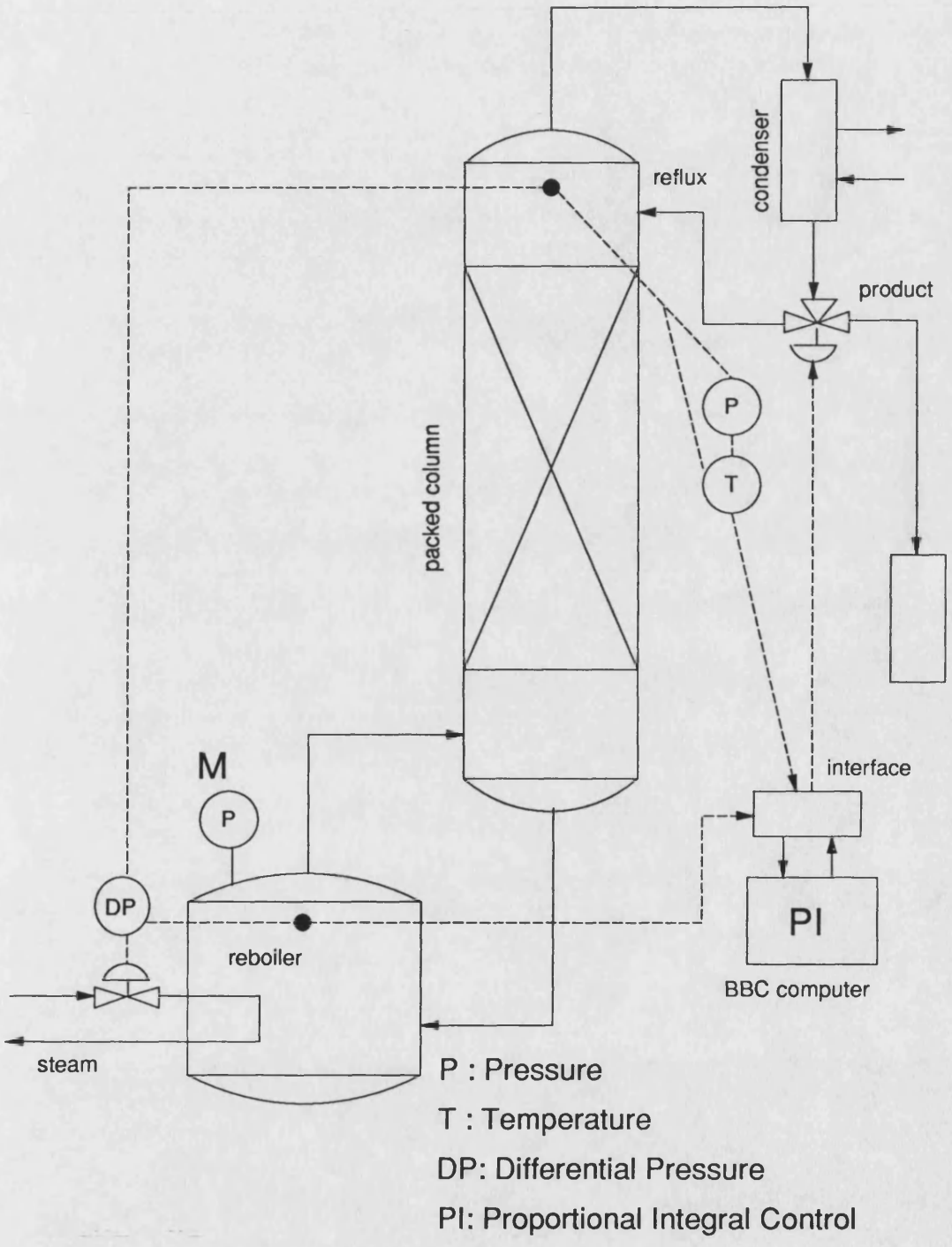


Figure 3.6 Schematic diagram of control system

Before composition can be successfully controlled, it must be measured, but composition is a difficult property to measure, and on-line analyzers tend to be complex, expensive and requiring of maintenance. The principal disadvantage of these analyzers would be the dead time encountered in various places such as the dead time that exists within the analyzers themselves. Both analysers used, refractometer for the binary system and gas chromotograph for the ternary system, must be mounted in protected locations and have a prepared sample brought to them. The dead time (about 5-10 minutes) they introduce is very long for control purposes. For these reasons, composition analyzers are not preferred and an indirect composition estimation technique was therefore used. For binary and almost binary systems when the molecular weights of the components are substantially different, indirect composition estimation may give good results. For close control of multicomponent systems, however, analyzers are virtually mandatory (Buckley et.al.,1985).

The output variable (overhead composition, OC) and the manipulated variable (reflux ratio) are calculated from correlations based on overhead temperature (T9-Top column thermocouple readings) and pressure (APT-Absolute Pressure Transducer readings) data. Vapour-liquid equilibrium relationship was used for indirect overhead vapour composition estimation as a function of overhead temperature and pressure for the binary system (see equation OC-Eq in Figure 3.7). To test the validity of the correlation, 19 liquid samples were taken from the reflux line and analysed using the refractometer. The compositions of these samples were also correlated empirically as a function of the overhead temperature (see equation OC-Samp). Figure 3.7 shows that results obtained from the analyzer are in good agreement with the overhead vapour composition correlation from the equilibrium data and with condensed liquid reflux line empirical

composition correlation. It should be noticed that composition values were plotted against the time scale. The reason for this representation is to emphasis the time lag between the temperature and pressure measurement point and the sampling point. Overhead samples had to be taken from the reflux line as subcooled liquid because of safety reasons, but this causes a time lag of 8.5 minutes. In other words, the sampling point produces data for the mixture that was on the top of the column 8.5 minutes ago. However, compositions could be estimated on both points from two correlations given in Figure 3.7, equilibrium correlation was used for the control experiments to eliminate the time lag. Composition estimation correlations derived for the ternary system are given in Appendix 10.

The dynamic performance of the column may be significant if the overhead concentration must be held constant. In that event, the reflux ratio requires continuous adjustment. Depending on the internal construction, the liquid content loading in the column and therefore the dynamic characteristics of the plant may change even at the constant vapour load, and the reflux control becomes sensitive. To maintain an even load in a batch column, the controls must provide constant pressure at both the top and the bottom of the column (Perry and Green, 1982).

The pressure of the reboiler was controlled manually using a pressure regulator (see PR in Figure 3.1) that regulated the amount of steam. The process pressure in the reboiler was observed with a pressure gauge installed at the top of the reboiler. The differential pressure should be maintained constant. However, manual control of the steam pressure regulator proved to be inefficient for small differential pressure changes. To minimize usage of PR for small variations, a normally close solenoid valve (see S1 in Figure 3.1)

was installed on a longer coil line of the steam inlet. If the current differential pressure is lower than the set point, indicating the boilup rate is lower than the desired level, S1 is energized to open to increase heat transfer area of the steam coils.

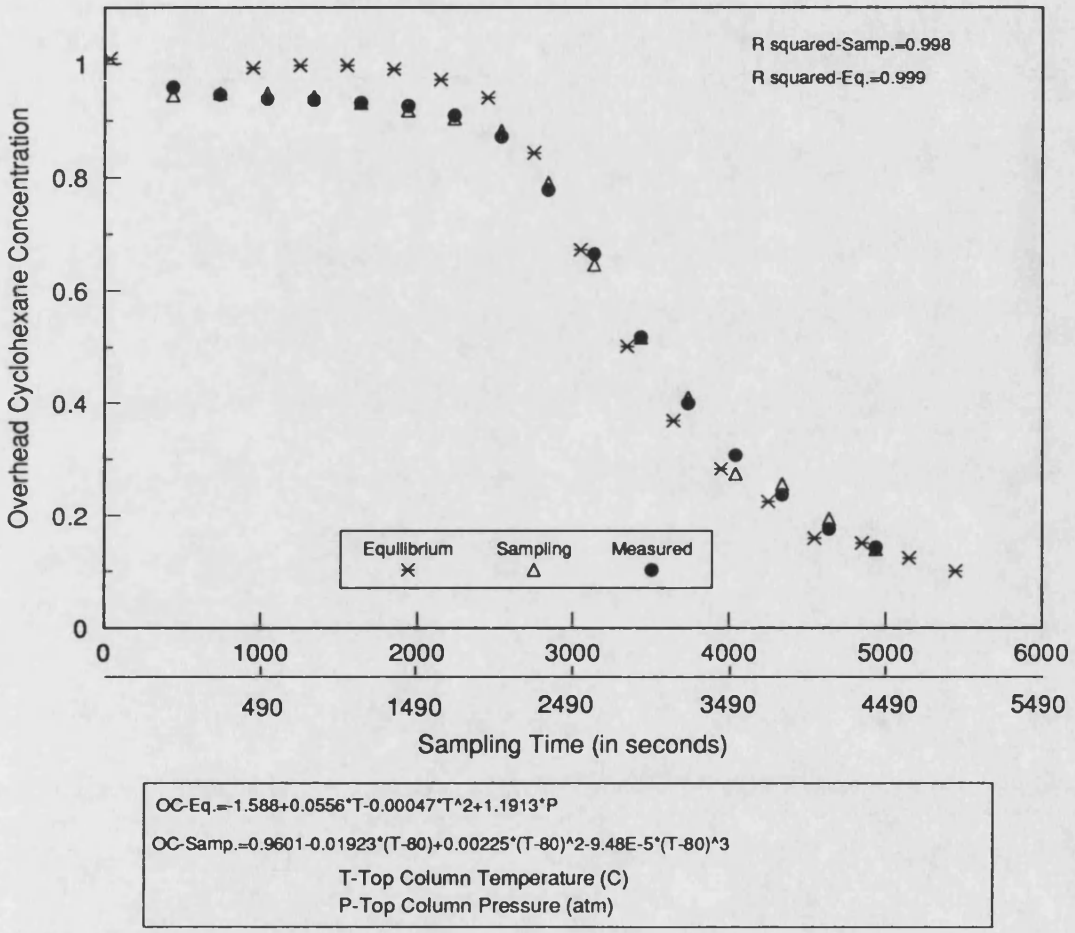


Figure 3.7 Indirect composition estimation correlations on the top of the column and the sampling point for the binary system.

3.7.1 System Identification

The control parameters used for the control algorithm are summarized in Table 3.7. The control parameters were obtained through the experiments.

Table 3.7 Control System Description

Sample Interval	5 sec.
Control Variable	Internal Reflux Ratio
Output Variable	Overhead Concentration as Mol Fraction
Disturbance	Pressure Drop in the Column
Time Constant	20 intervals (100 sec.)
Gain	6.5E-03
Dead Time	10 intervals (50 sec.)

For an effective controller design, the sample interval size should be short enough to define system dynamics adequately. For a first order system with a dead time, if the dead time τ_d is of the same order of magnitude as the time constant τ_p, then the sample interval T is recommended that should be selected equal to $0.1\tau_d$ or $0.1\tau_p$, whichever is smaller. If τ_d is much smaller than τ_p, the dead time is neglected and $T=0.1\tau_p$ is taken (Stephanopoulos, 1984). The control sample period was taken as 5 seconds.

System identification may be defined as the determination of a model that adequately characterizes the dynamic performance of the system. A dynamic model can be developed theoretically and/or experimentally. The theoretical approach, based on fundamental physical and chemical laws, seeks to describe the time dependent behaviour of the system. However, as modelling of the packed batch distillation column is the main

objective of this work, its simulation requires an excessive time and memory. The BBC computer that was allocated for the control system is not capable of approaching the problem theoretically.

For a system available for experiment, the empirical approach is essential. It is also used when the system is thought to be too complex to model from fundamental laws, or when the values of some parameters are unknown. Because, dynamic experiments would be used to confirm the prediction of a theoretical mathematical model, an empirical system identification approach was applied in this work.

Experimental identification of system dynamics has been an active area of research for many years. A number of techniques has been developed (Shina and Kuszta, 1983). The most used method is the step response. This is the most direct way of obtaining an empirical model of a process and finding the parameters, such as, dead time, time constant and process gain, that fit the experimentally obtained step response. A step change is made in the manipulated variable and the response of the process output is recorded. The resulting input-output data are fitted to a first order model containing dead time.

Identification experiments were carried out with the binary system (Cyclohexane-Toluene) and manual control of steam pressure. Data from identification experiments indicated that the system is adequately defined as first order plus dead time.

Figures 3.8 and 3.9 show the responses of product cyclohexane concentration in several step changes in reflux flow. Disturbance is the pressure drop across the column in [mm Water] (divided by 10 to be able to show in the same scale), concentration is the mol fraction of cyclohexane and reflux ratio is the inner reflux ratio=L/V in these figures.

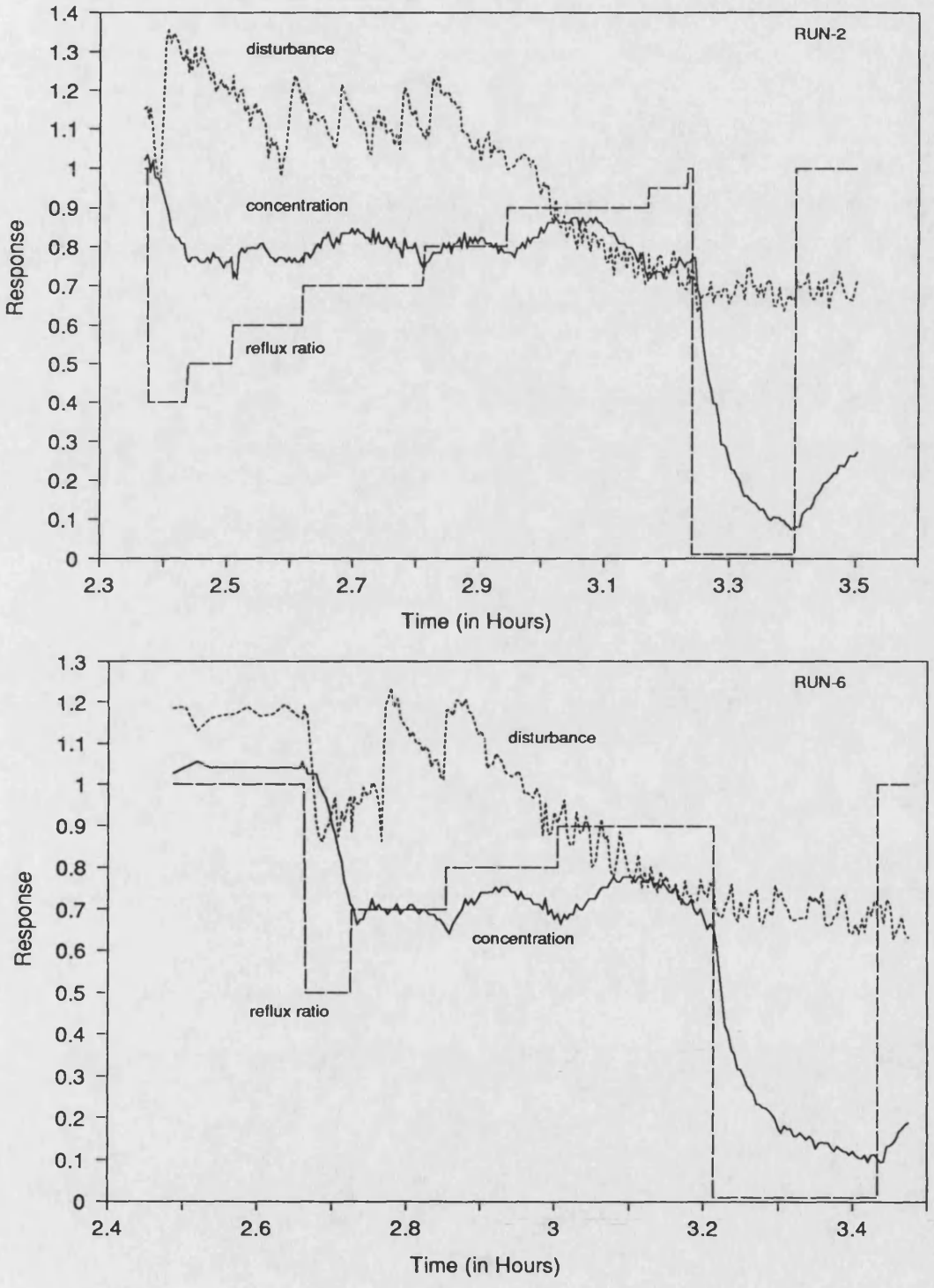


Figure 3.8 Experimental results for identification.

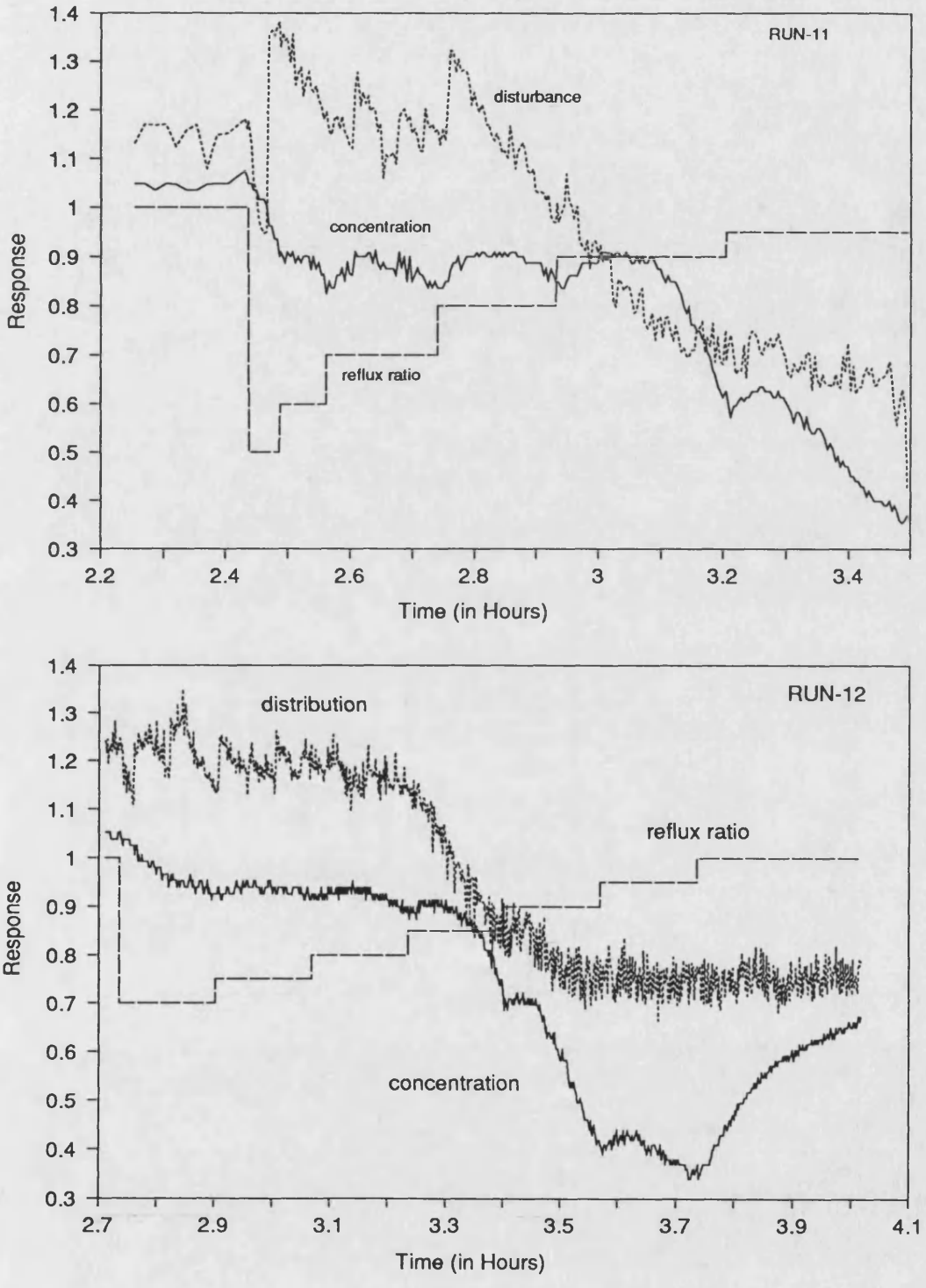


Figure 3.9 Experimental results for identification.

An average response to a step change in reflux ratio is plotted in Figure 3.10. In this figure, the circles show the experimental data and the line denotes the model fitting curve.

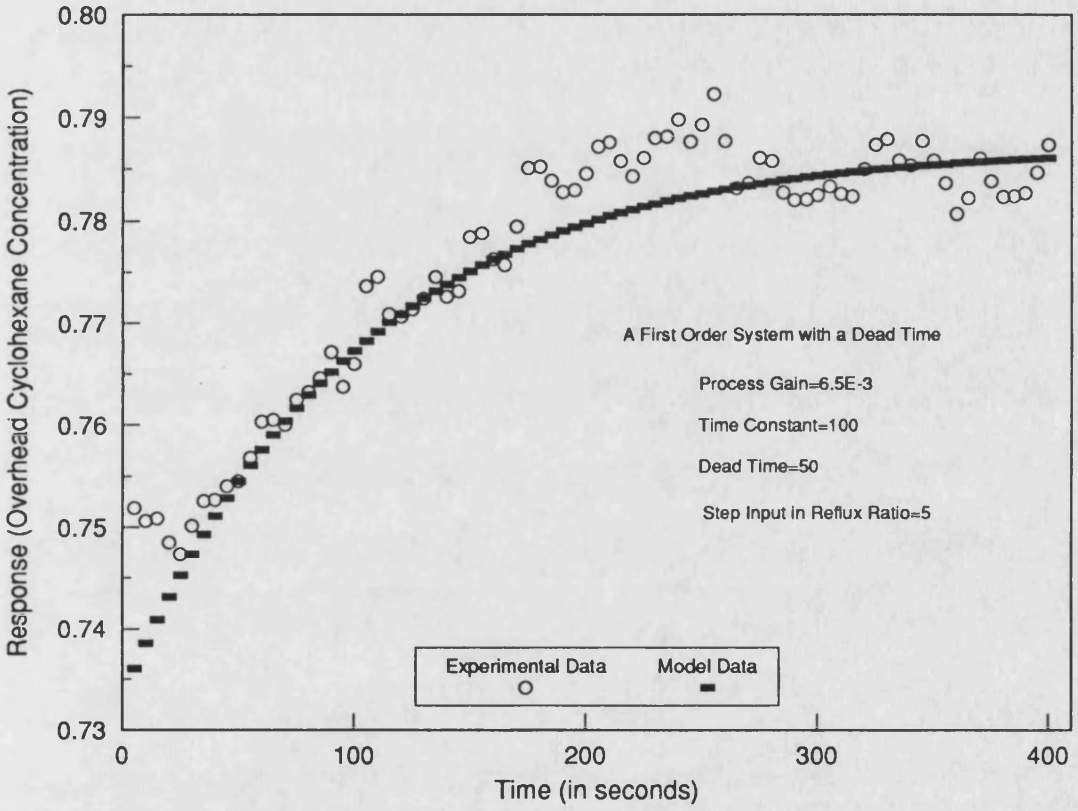


Figure 3.10 Response of step change in reflux ratio.

From the experimental data of Figure 3.10, it can be seen that the system can adequately be modelled as first order. The line shows the first order model with a time constant of 100 (20 time intervals) and steady state gain of 6.5E-3. The model curve and experimental data are in close agreement. The time constant τ_p of a process is a measure of the time necessary for the process to adjust to a change in its input. The value of the response reaches 63.2% of its final value when the time elapsed is equal to one time constant, τ_p.

The steady state or static gain K_p gives the resulting change in the output steady state to any step change Δ(input), in the input, by $\Delta(output) = K_p\Delta(input)$. The dead time τ_d defines the time delay between the input and the output. The transfer function for this system is then given by Eq.(3.1):

$$G_p(s) = \frac{\overline{OC}(s)}{\overline{R}(s)} = \frac{K_p e^{-\tau_d s}}{\tau_p s + 1} = \frac{6.5E-3e^{-50s}}{100s+1} \tag{3.1}$$

Inverting Eq.(3.1), when the step change in R(t) was of magnitude A=5, the response would be

$$OC(t) = AK_p\left(1 - e^{-(t-\tau_d)/\tau_p}\right) = 32.5E-3(1-e^{-(t-50)/100}) \tag{3.2}$$

3.7.2 Controller Tuning

Sampled data controllers can be designed in the same way continuous controllers are designed (Luyben, 1990).

Ziegler-Nichols technique that is based on frequency response analysis was used for the PI controller parameter tuning. The analysis of stability characteristics of a system containing dead time is conveniently handled with the aid of frequency response diagrams. Classically, frequency response refers to the steady state portion of the response of a process to a sinusoidal input. It is characterized by two parameters, amplitude ratio and phase angle, both being functions of the frequency of the sinusoid. The preferred approach for the determination of the frequency response of a process to a sinusoidal input is to substitute jw for s in the transfer function and determine the

amplitude and the angle of the resulting complex number. It turns out that the amplitude of the complex number equals amplitude ratio and the angle of the complex number equals the phase angle (Deshpande, 1985).

The Bode diagrams constitute a convenient way to represent the frequency response characteristics of a system. The Bode diagrams consist of a pair of plots showing how the amplitude ratio and the phase shift varies with frequency.

For a first order system with dead time, amplitude ratio and phase lag are given

$$AR = \frac{K_p}{\sqrt{1+\tau_p^2\omega^2}} \qquad (3.3)$$

$$\Phi = \tan^{-1}(-\tau_p\omega) - \tau_d\omega \qquad (3.4)$$

The Bode diagram for the open-loop transfer function that is given by Eq.(3.1) can be constructed easily and is shown in Figure 3.11. Where the phase lag is $\Phi = -180°$ the frequency is called the crossover frequency and from Figure 3.10 $\omega_{CO} = 3.568642 rad/\sec$ was found. M=0.001724 is the amplitude ratio at the crossover frequency. According to the Bode criterion if M<1 then the closed loop system is stable. Gain margin=1/M is a measure of how far the system from the instability. For a stable system gain margin is above the value of 1 and in this case it is 580.

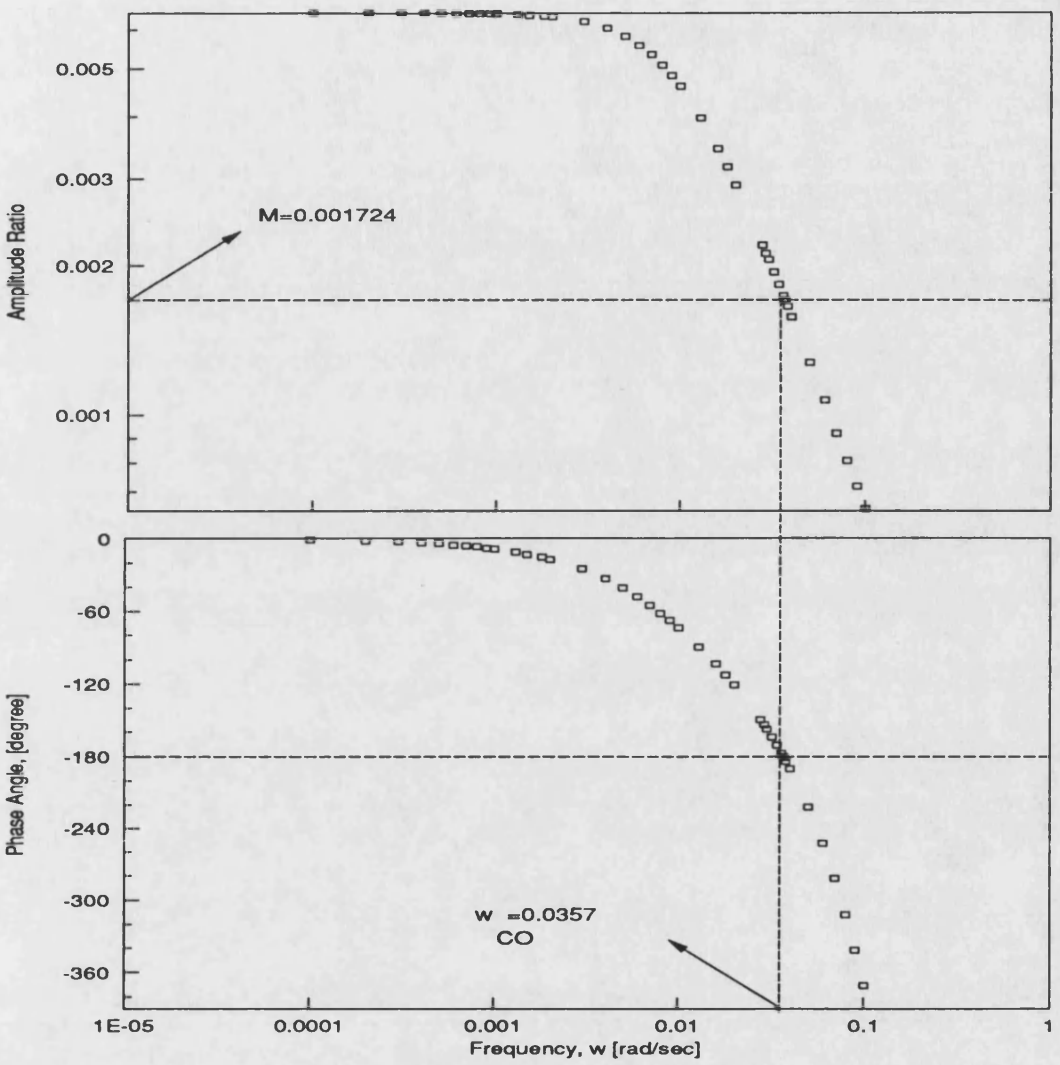


Figure 3.11 Definition of gain and phase margins using Bode diagram.

Using Ziegler and Nichols recommended settings for a PI controller, K_c=264 and τ_I=146 were calculated (Stephanopoulos, 1984).

Consequently, the control action of a digital PI controller is determined by the following discrete-time model:

$$R(nT) = R_s + K_c \varepsilon(nT) + \frac{K_c T}{\tau_I} \sum_{k=0}^{n} \varepsilon(kT) \tag{3.5}$$

$$\varepsilon(nT) = OC_{SP}(nT) - OC(nT) \tag{3.6}$$

where R_s is the controller output signal (inner reflux ratio=L/V) when the error is zero. It was taken as 0.70 or 0.80 in control experiments. ε(nT) is the discrete time error. In every sampling period a sampled value of the process output enters the computer. Let OC(nT) be the sampled value at the nth sampling instant. OC(nT) is compared to the set point value at the same instant and yields the value of ε(nT). Numerical results of control experiments will be discussed thoroughly in Chapter 4.

3.8 Conclusion

As stated clearly by Hitch and Rousseau (1988) and Jang (1990) there is a big gap in the literature for the experimental data of the multicomponent packed batch distillation. It is believed that the present experimental work will make a contribution in this area. Nineteen experiments were carried out using the experimental equipment (photograph given in Appendix 3). Binary and ternary mixtures were used in these experiments and the performance of the column was tested for different experimental initial conditions and operation. The experimental data are given in Tables 3.6 and 3.7.

CHAPTER 4

VERIFICATION OF THE MODEL USING EXPERIMENTAL RESULTS

The preferred method for testing the validity of a simulation procedure is to compare the predictions of the algorithm with actual experimental results. This chapter presents verification of model results of PACBACDIS (introduced in Chapter 2) with the experimental results which were obtained for a binary and a ternary system using a packed batch distillation column. Experimental data for 19 experiments are given in Chapter 3 in Table 3.5 for binary systems and in Table 3.6 for ternary systems. Experiments are defined as RUN-# and Table 3.5 and Table 3.6 can be consulted for the experimental conditions and results.

The experimental data consist of the temperature profile of 5 different points in the packed section, top column pressure, pressure drop across the column and overhead composition of the mixture. Data are also available for the reboiler and condenser heating and cooling stream input and output temperatures respectively. The top column pressure is the only experimental data which is used as an input to PACBACDIS. The rest is also obtained as output data of PACBACDIS during simulation runs. This gives enough range of data for a model verification.

4.1 Number of Nodes Considered

As explained in Chapter 2, the numerical solution procedure of the model equations of partial and algebraic equation sets using Finite Difference discretization considers grid nodes in the packed section. The accuracy of the solution very much depends on the number of nodes taken. In this section, a study was carried out in order to find the number

of nodes which should be taken in the packed section to increase the accuracy of the model results. Figure 4.1 shows the effect of number of nodes on top column liquid mol fraction values and calculated data is given in Table 4.1 for the top column. 33 nodes were chosen from the list of nodes studied for PACBACDIS simulation runs.

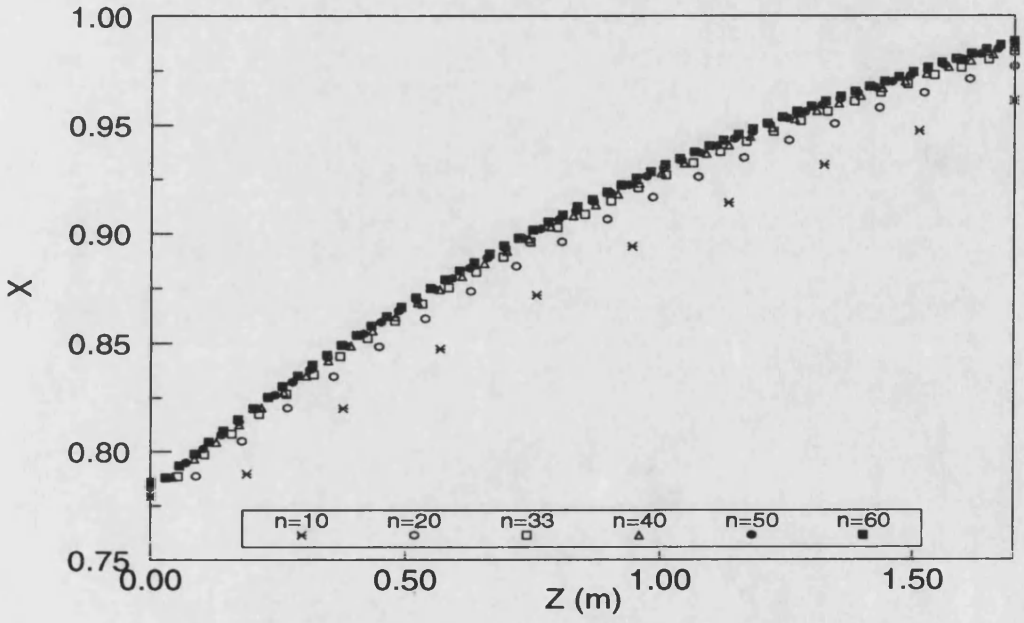


Figure 4.1 Effect of the number of elements taken in packed section on total reflux steady state calculation results for RUN-1 case.

Table 4.1 Calculated Top Column Liquid Mol Fraction Values For Different Δz.

Number of nodes	Δz (m)	x (Z=1.7 m)	Error, %
10	0.188889	0.961230	2.77
20	0.089470	0.977232	1.15
33	0.053125	0.983804	0.49
40	0.043590	0.985640	0.30
50	0.03469	0.987402	0.13
60	0.028814	0.988637	0.00

4.2 Parametric Study

The model of batch packed distillation column given in Chapter 2 consists of some parameters and in order to carry out the model calculations, these following parameters had to be evaluated: mass transfer coefficients, k_y, k_x, axial dispersion coefficients, D_v, D_l and holdups h_v and h_l of vapour and liquid phases respectively.

For this evaluation, the methods in the literature (refer to Table 2.2) is not satisfactory because they are mainly empirical models which were derived for different conditions. For example, the main parameter for the simulation of the column behaviour is the mass transfer coefficients between the liquid and gas phases. The correlations which are given in the literature for the calculation of these coefficients are defined by means of a simple model (Onda et.al., 1968).

In order to obtain these parameters to define the experimental conditions better, a systematic study of the sensitivity of variation of the model parameters was carried out. These variations were achieved by multiplying the parameters which were obtained in the literature by constant factors, c, which will be used as references (parameter-used=c*parameter-calculated). This study was conducted for the steady state model results and calculated model variables, liquid and vapour phase mol fractions, temperatures and liquid and vapour flow rates were examined. Behaviour of model variables against model parameter variations are illustrated by Figures from 4.2 to 4.7. Initial conditions and experimental data of experiment RUN-1 (refer to Table 3.5) were used for comparison.

For all the parametric study runs, except the parameter under consideration, the values of the parameters used are as follows:

Vboil = 0.25 [kmol/h] (constant through the run)
G_c = 0.01 [m^3 liquid/m^3 condenser-reflux drum]
h_l = 0.01 [m^3 liquid/m^3 column packed section]
h_v = 0.01 [m^3 vapour/m^3 column packed section]
c for k_x = 15.2
c for k_y = 0.066
c for D_l = 1.0
c for D_v = 1.0

The variation of k_y has a great effect on the profile of model variables as seen Figure 4.2. When k_y increases, the model results approach the experimental results. Figure 4.2.(d) shows the variations of temperature profiles of the model results and experimental data.

It should be noted that the vapour phase mass transfer coefficient has a much greater effect than the liquid phase mass transfer coefficient on the mol fractions. Although the effect of liquid mass transfer coefficient variations on bulk phase mol fractions is negligible (Figure 4.3.(b) and (c)), interfacial mol fractions change significantly with these variations. As a result, the temperature profile changes.

The calculated results are not very sensitive towards the variation of D_l as seen in Figure 4.5. However, the results are more sensitive towards the variations of D_v (Figure 4.4). Figures 4.2 and 4.4 show that the effect obtained by increasing D_v is opposite to the one achieved by increasing k_y.

The effect of the variations of liquid and vapour holdups, h_l and h_v, on the steady state model results is very much on the flow rates. h_v could play an important role

especially for high pressure columns. As long as the values of the liquid and vapour holdups are between boundaries, the effect of their variations is not very important. However, when they exceed the boundaries, the results are suddenly not acceptable (Figures 4.6 and 4.7).

As a result of the parametric study, It was concluded that the following coefficients should be used for the simulation runs:

h_l = 0.01 [m^3 liquid/m^3 column packed section]

h_v = 0.0001 [m^3 vapour/m^3 column packed section]

c for k_x = 17.5

c for k_y = 0.067

c for D_l = 0.0001

c for D_v = 0.0001

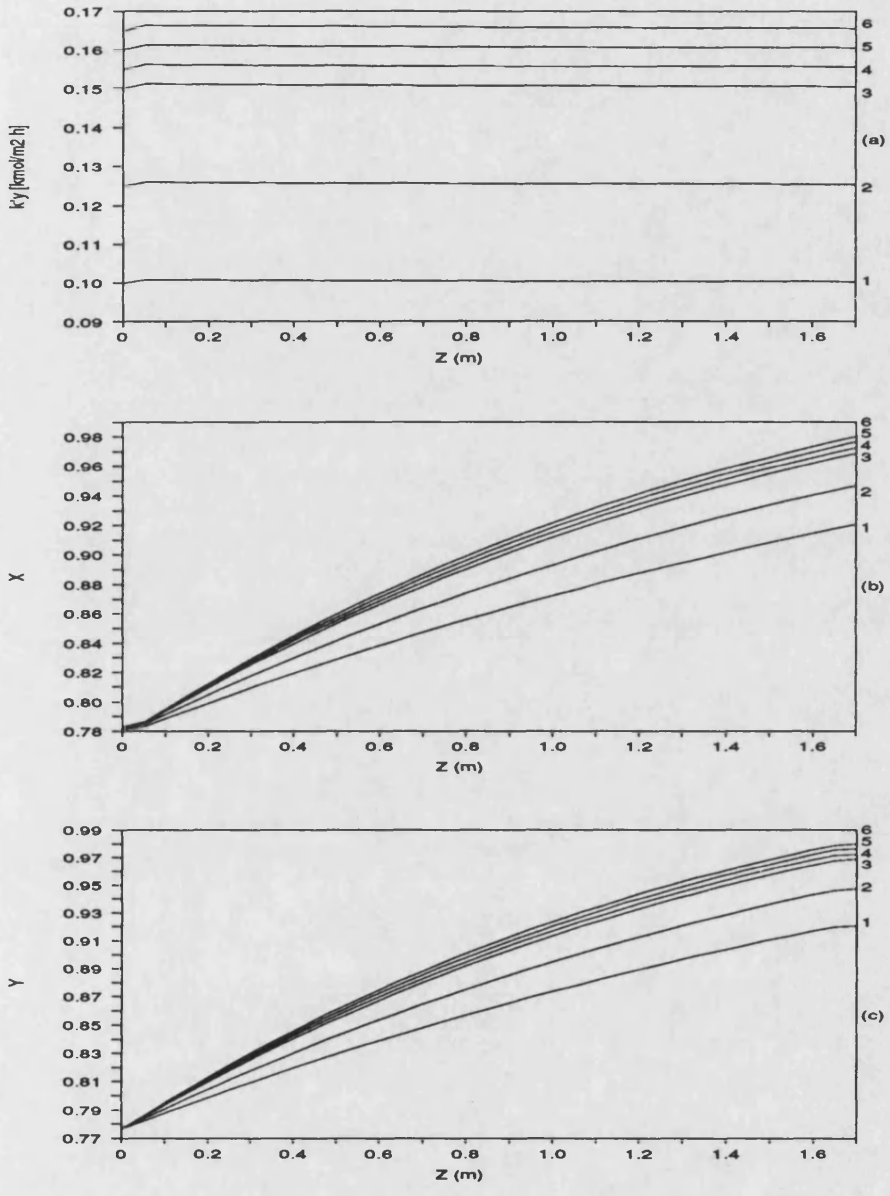


Figure 4.2. Effect of gas mass transfer coefficient variations on model variables of steady state model for RUN-1 condition, $c=ky/ky_{calculated}$

(a) 1-c=0.04; 2-c=0.05; 3-c=0.06; 4-c=0.062; 5-c=0.064; 6-c=0.066

(b) calculated X profile in the packed section

(c) calculated Y profile in the packed section

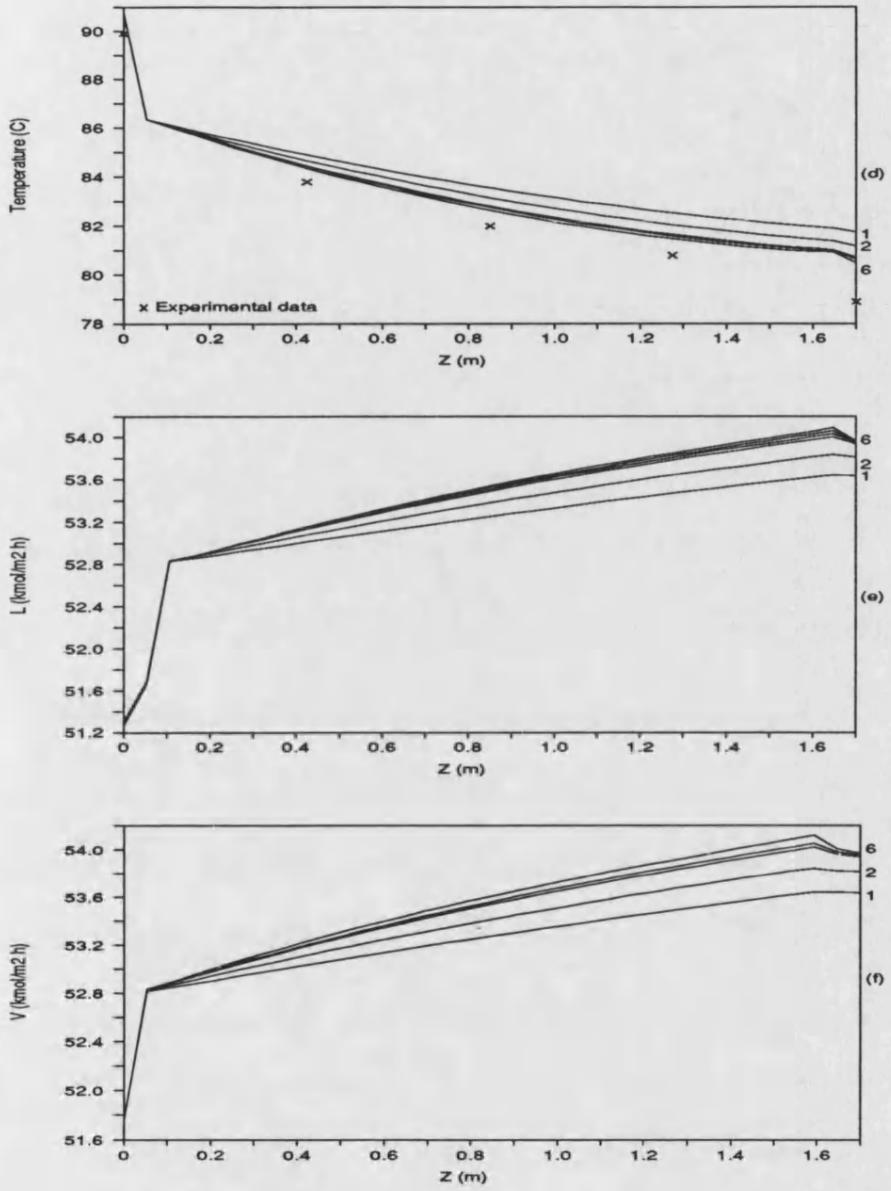


Figure 4.2.

(d) calculated Temperature profile in the packed section

(e) calculated liquid flow rate profile in the packed section

(f) calculated vapour flow rate profile in the packed section

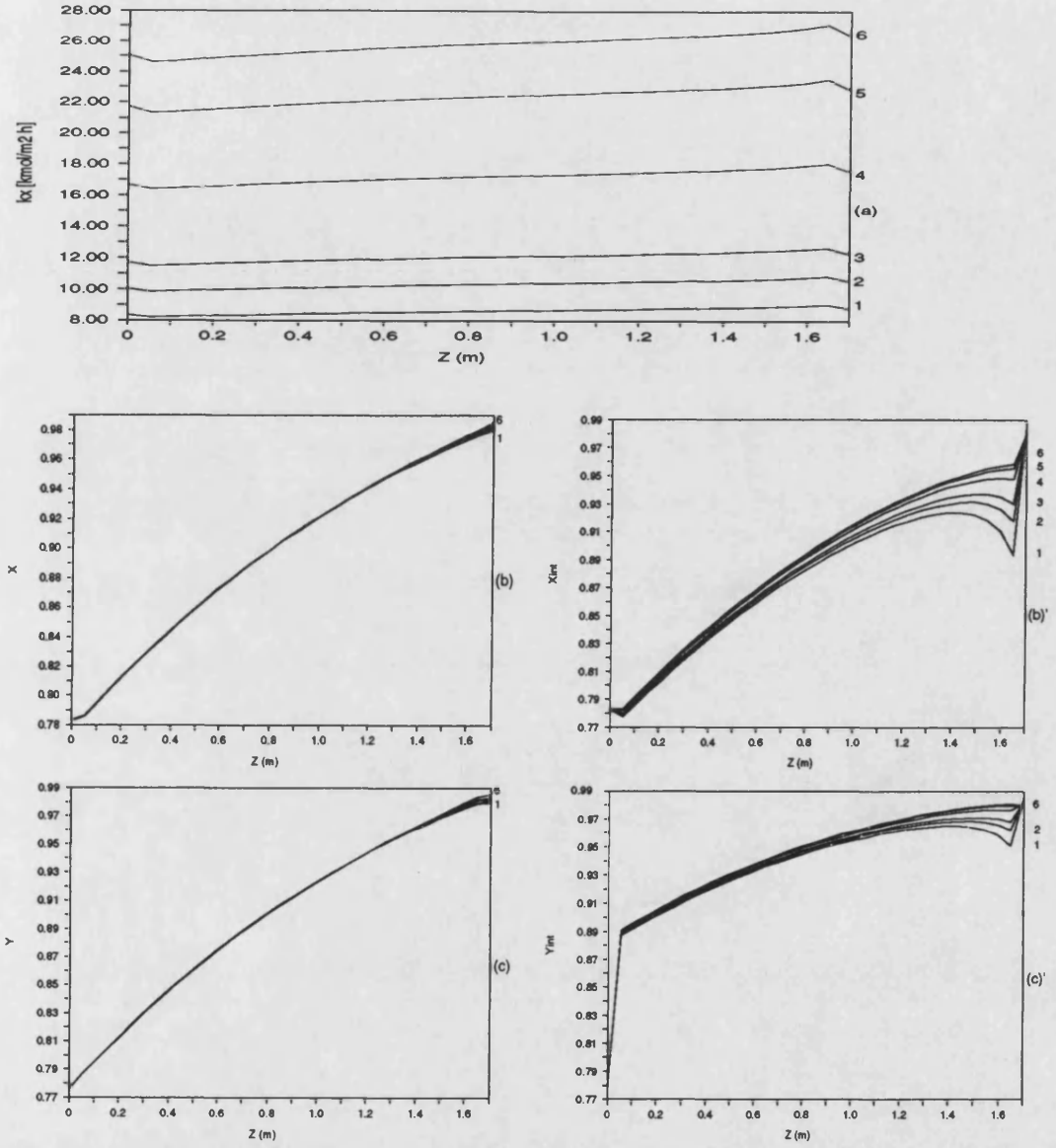


Figure 4.3. Effect of liquid mass transfer coefficient variations on model variables of steady state model for RUN-1 condition, $c=kx/k_{xcalculated}$

(a) 1-c=5; 2-c=6; 3-c=7; 4-c=10; 5-c=13; 6-c=15

(b),(b)' calculated X and Xint profiles in the packed section

(c),(c)' calculated Y and Yint profile in the packed section

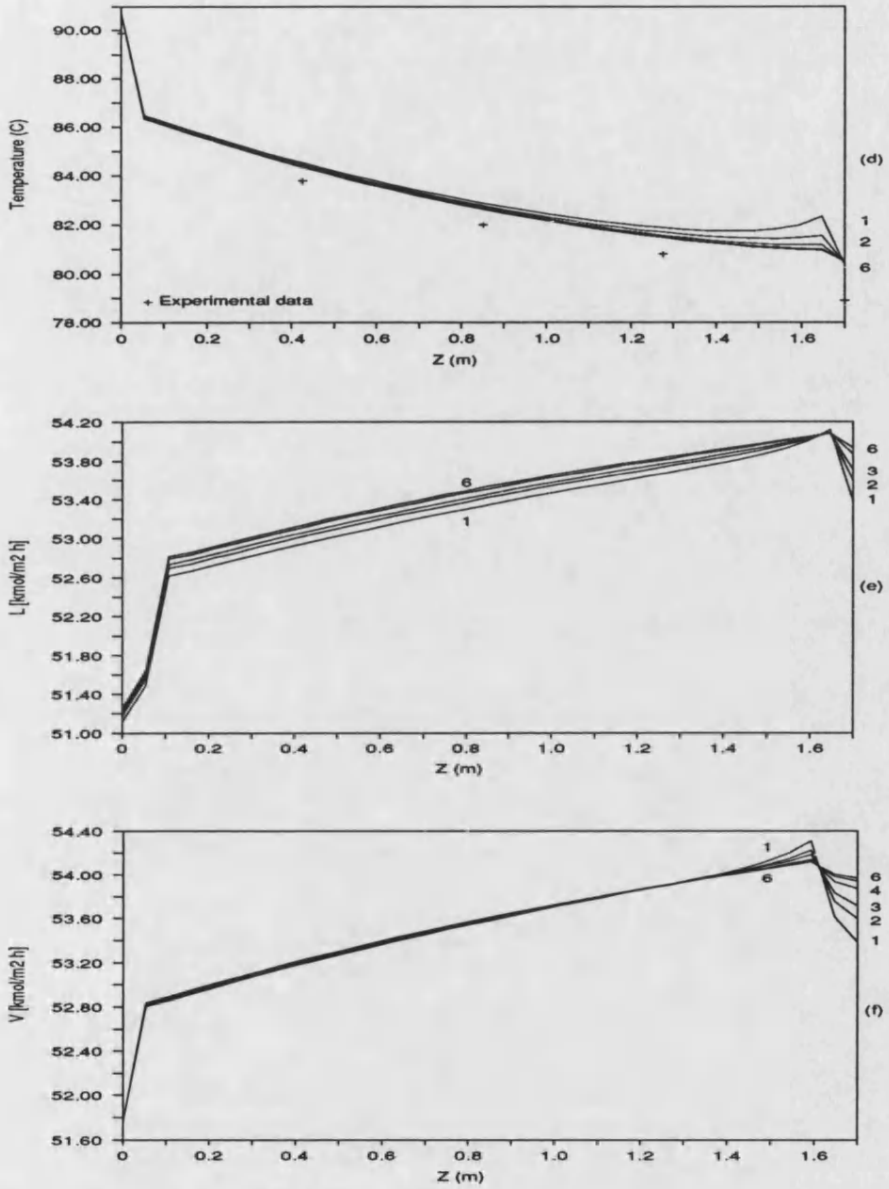


Figure 4.3.

(d) calculated Temperature profile in the packed section

(e) calculated liquid flow rate profile in the packed section

(f) calculated vapour flow rate profile in the packed section

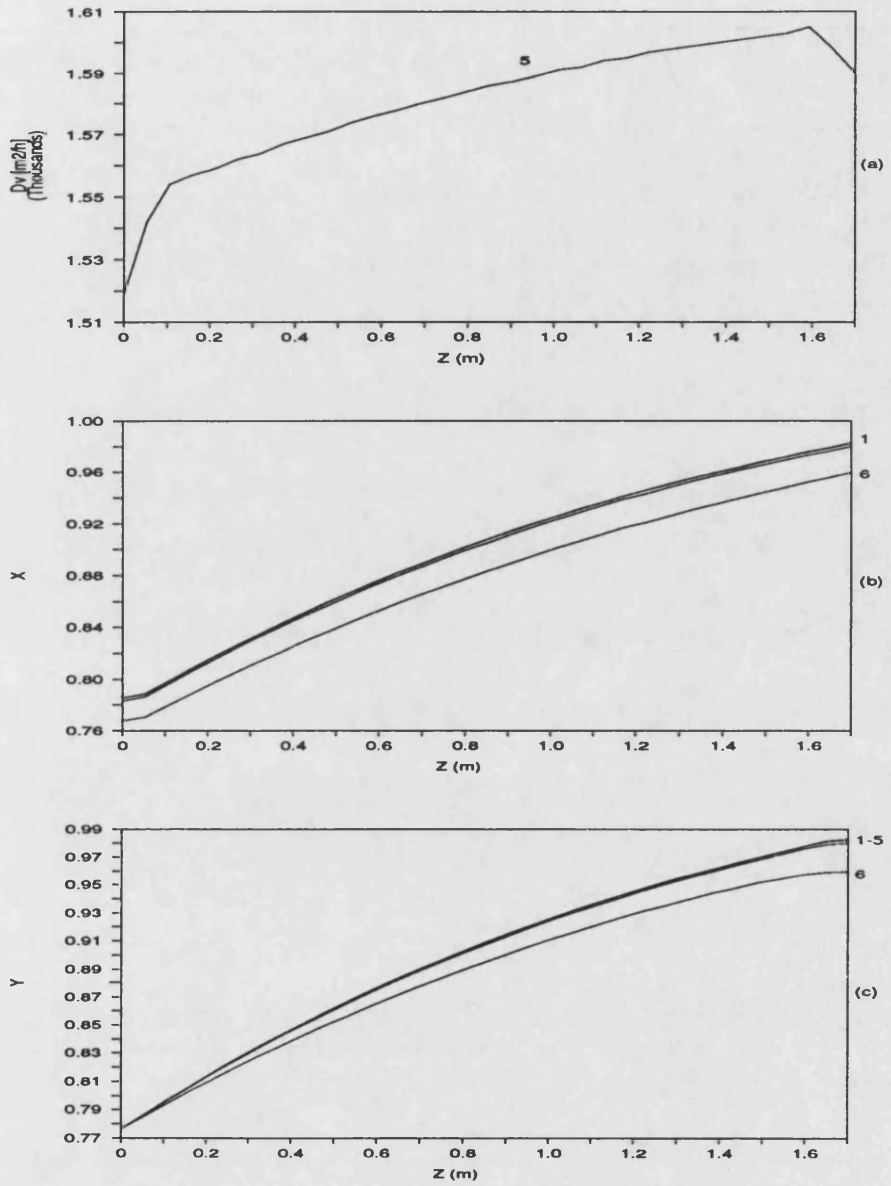


Figure 4.4. Effect of vapour phase axial dispersion coefficient variations on model variables of steady state model for RUN-1 condition, $c=Dv/Dv_{calculated}$

(a) 1-c=0.00001; 2-c=0.001; 3-c=0.01; 4-c=0.1; 5-c=1; 6-c=10

(b) calculated X profile in the packed section

(c) calculated Y profile in the packed section

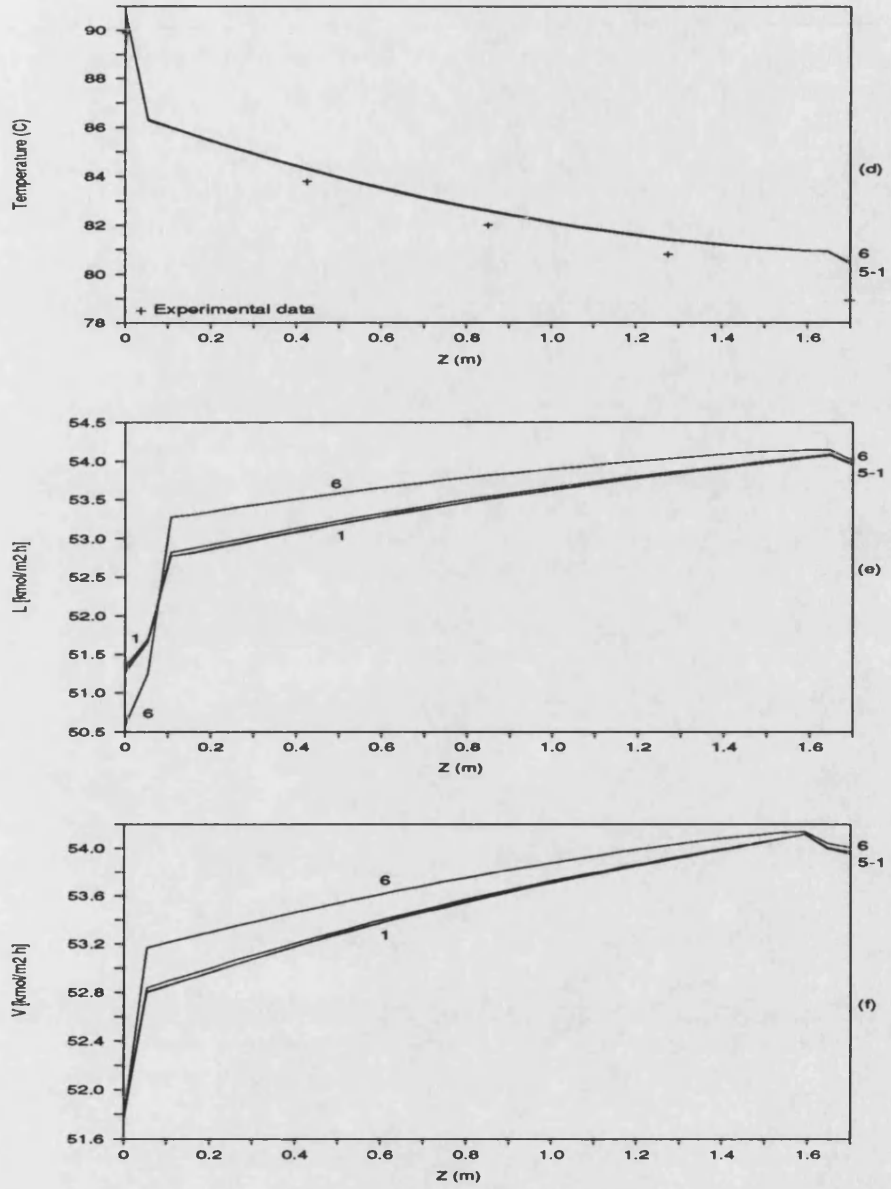


Figure 4.4.

(d) calculated Temperature profile in the packed section

(e) calculated liquid flow rate profile in the packed section

(f) calculated vapour flow rate profile in the packed section

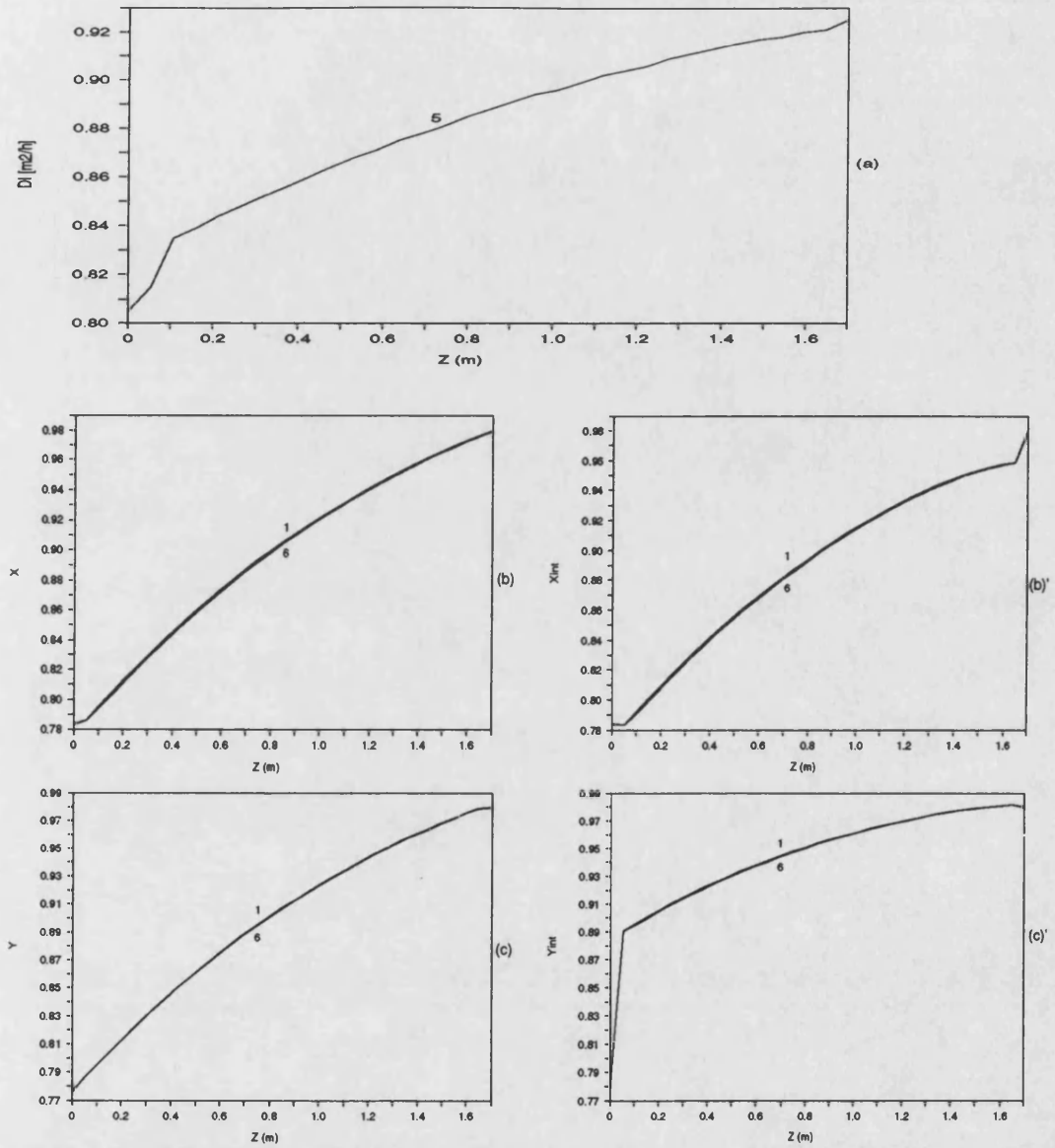


Figure 4.5. Effect of liquid phase axial dispersion coefficient variations on model variables of steady state model for RUN-1 condition, $c=Dl/Dl_{calculated}$

(a) 1-c=0.0001; 2-c=0.001; 3-c=0.01; 4-c=0.1; 5-c=1; 6-c=10

(b),(b)' calculated X and Xint profiles in the packed section

(c),(c)' calculated Y and Yint profiles in the packed section

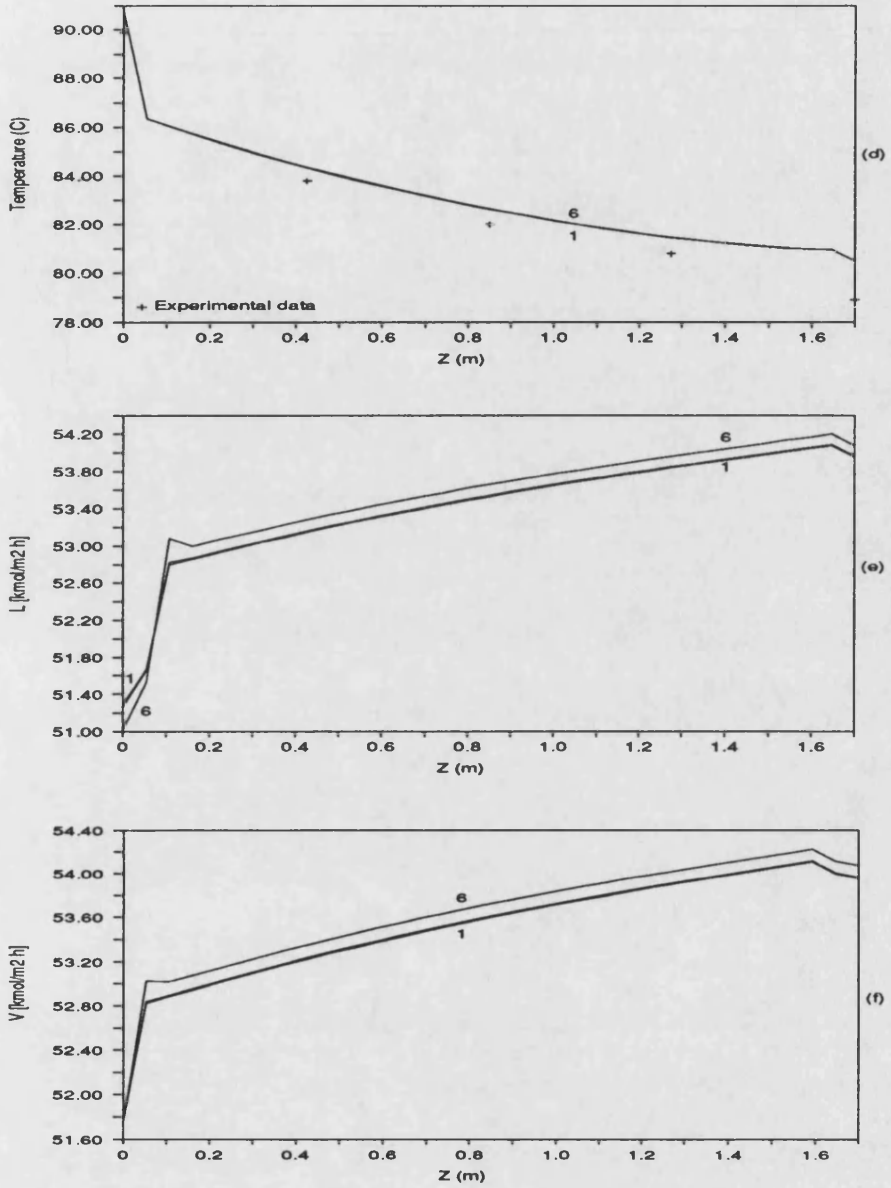


Figure 4.5.

(d) calculated Temperature profile in the packed section

(e) calculated liquid flow rate profile in the packed section

(f) calculated vapour flow rate profile in the packed section

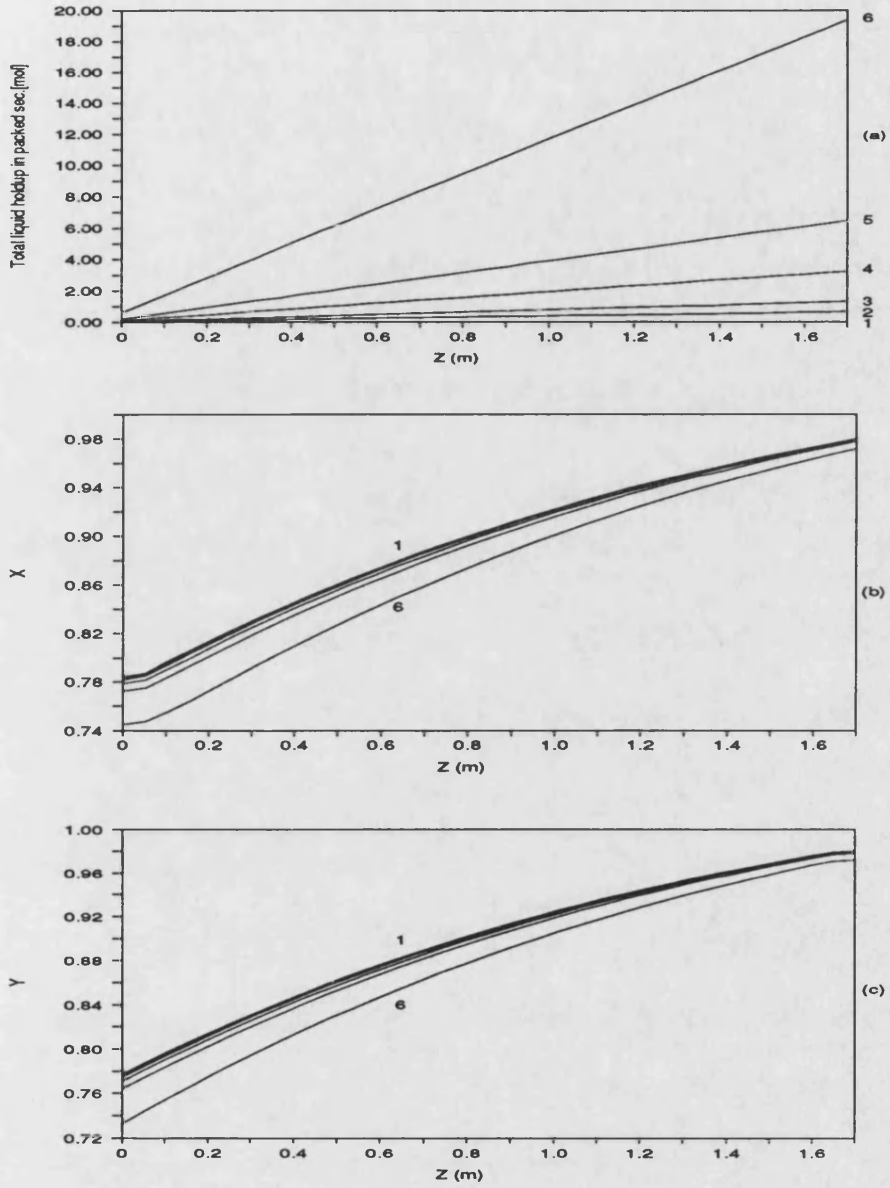


Figure 4.6. Effect of liquid holdup variations in packed section on model variables of steady state model for RUN-1 condition, hl as v%

(a) 1-0%; 2-1%; 3-2%; 4-5%; 5-10%; 6-30%

(b) calculated X profile in the packed section

(c) calculated Y profile in the packed section

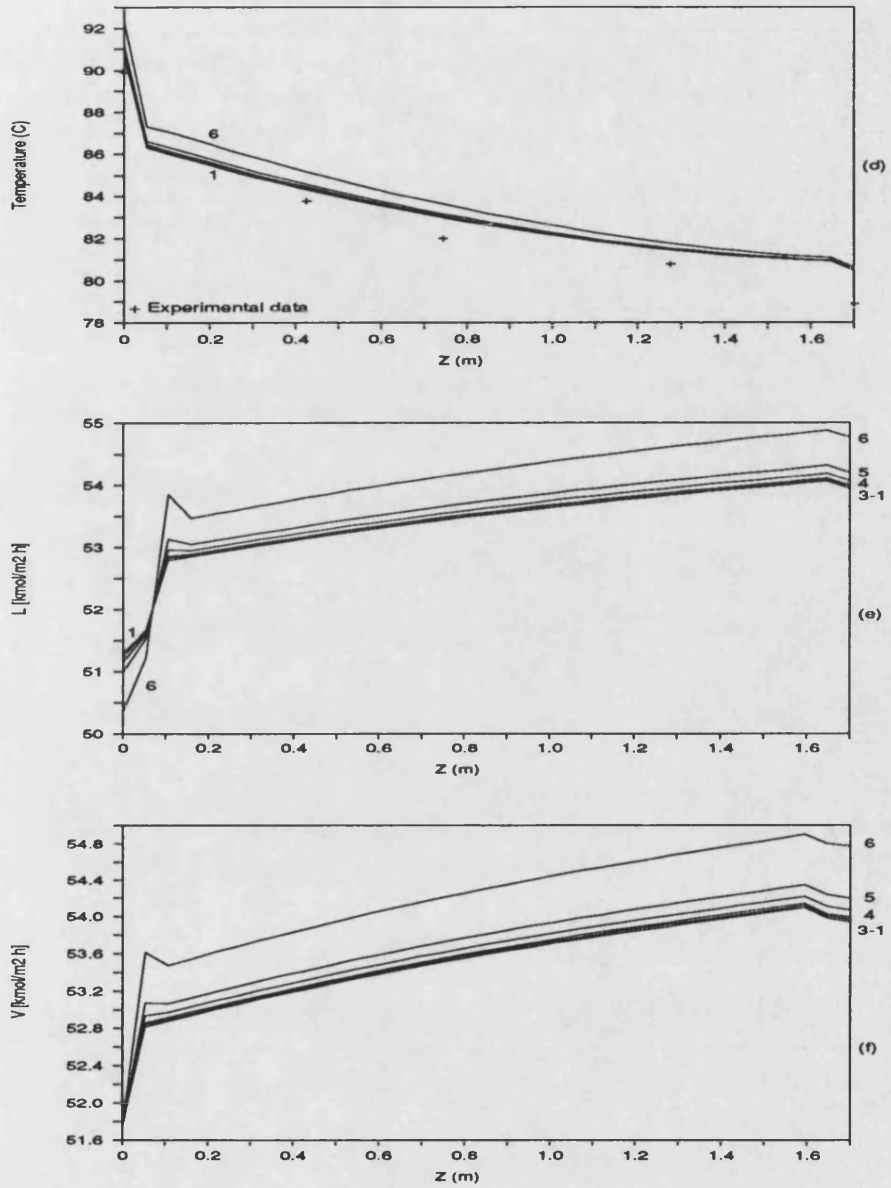


Figure 4.6.

(d) calculated Temperature profile in the packed section

(e) calculated liquid flow rate profile in the packed section

(f) calculated vapour flow rate profile in the packed section

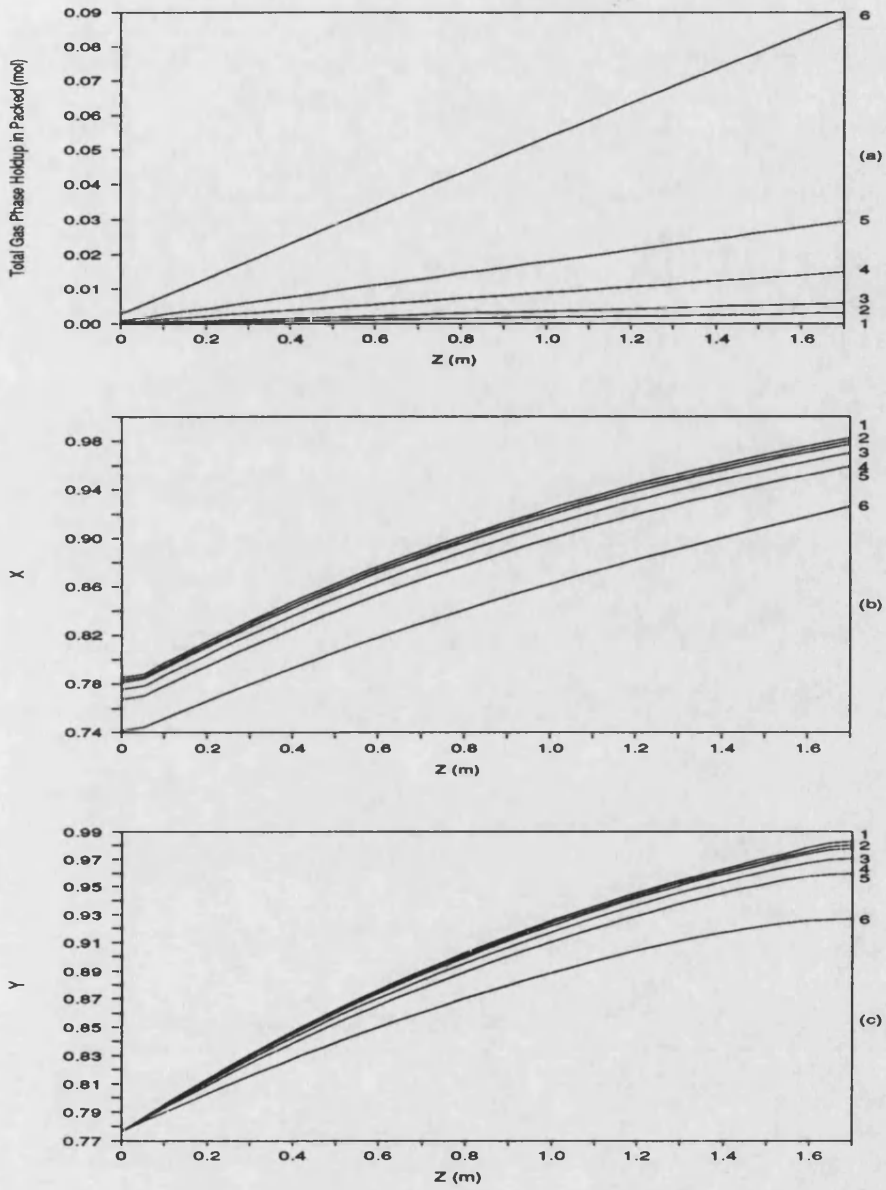


Figure 4.7. Effect of gas holdup variations in packed section on model variables of steady state model for RUN-1 condition, hv as v%

(a) 1-0%; 2-1%; 3-2%; 4-5%; 5-10%; 6-30%

(b) calculated X profile in the packed section

(c) calculated Y profile in the packed section

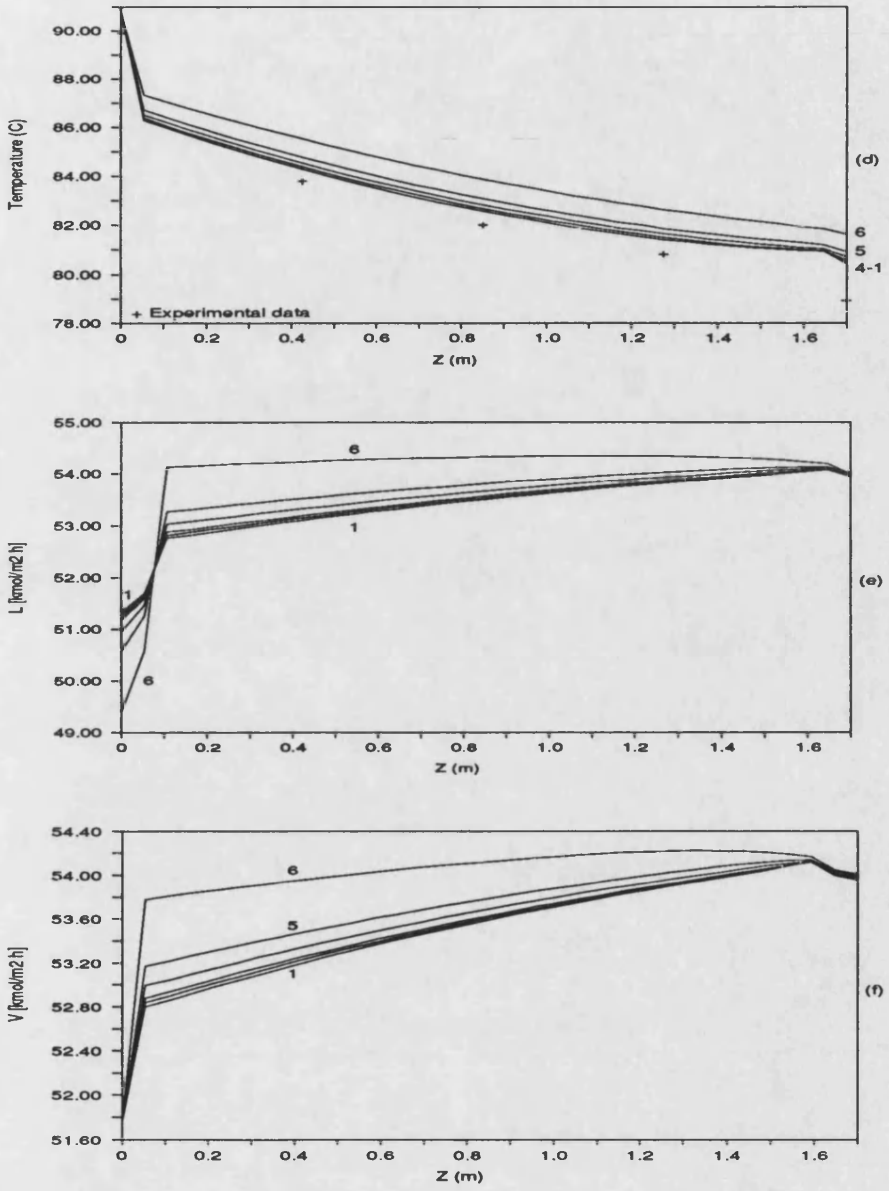


Figure 4.7.

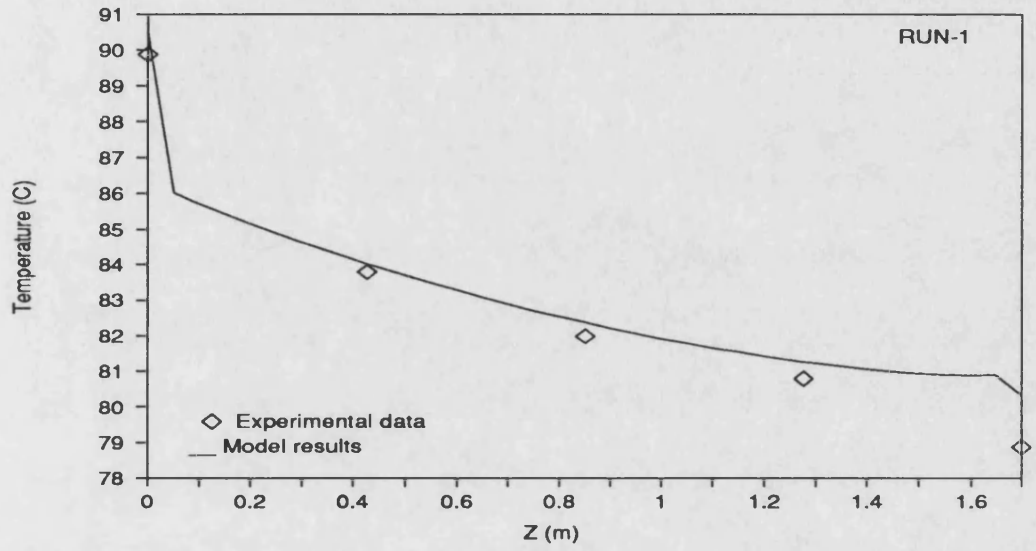
(d) calculated Temperature profile in the packed section

(e) calculated liquid flow rate profile in the packed section

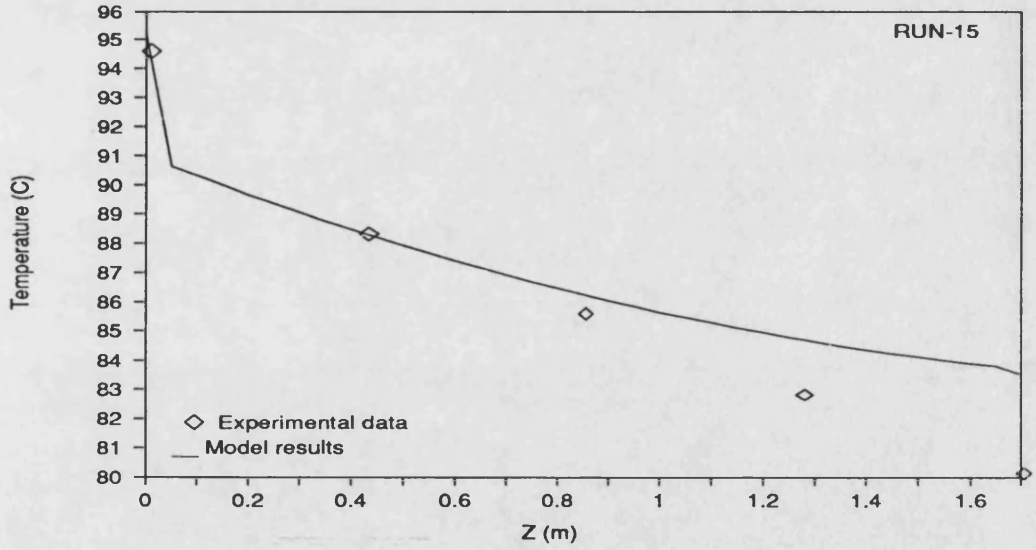
(f) calculated vapour flow rate profile in the packed section

4.3 Steady State Total Reflux Period

Figure 4.8 (a) and (b) illustrate the steady state total reflux temperature profiles in the packed section and the comparison between experimental data and model results for RUN-1 (binary system) and for RUN-15 (ternary system) using the above parameters.



(a)



(b)

Figure 4.8 Total reflux steady state temperature profile for RUN-1 and RUN-15

Experimental data and model results show a very good agreement in the packed section except in the vicinity of the top column. This discrepancy is caused by the subcooled reflux liquid that returns to the top of the column where the thermocouple was located for top column temperature measurements during experiments. The model assumes that the reflux liquid is at the same temperature of the vapour of top column and a total condenser is used. Obviously, the difference will be maximum for total reflux conditions that all subcooled reflux liquid returns to the top of the column.

The numerical solution of total reflux steady state period is very stable and converges to an error tolerance 1.0E-06. Figure 4.9 (a) and (b) illustrate typical convergence profiles of the RUN-1 (binary system) and RUN-15 (ternary system) calculations respectively.

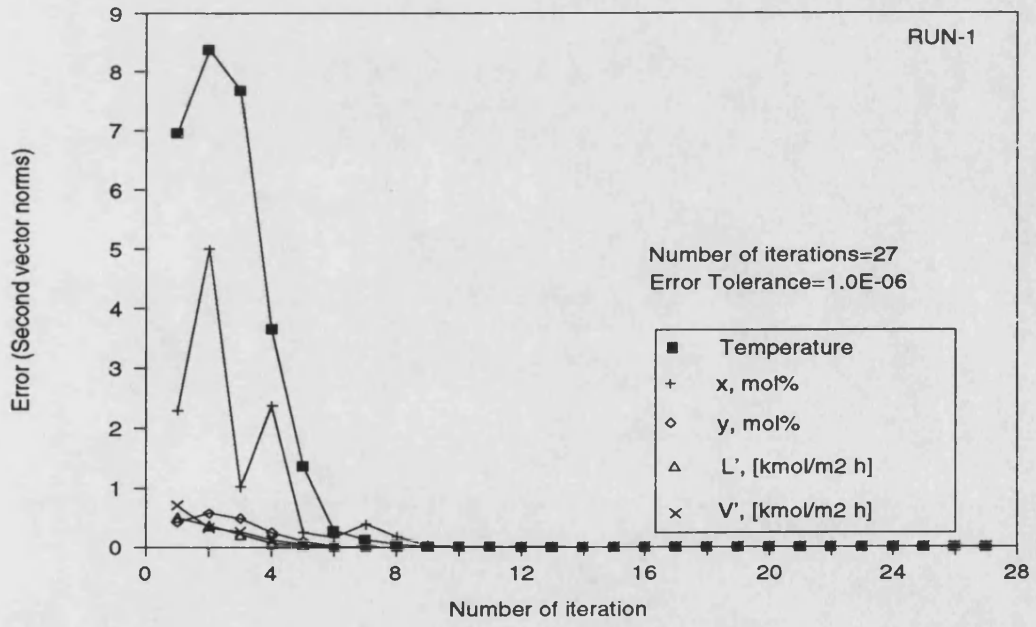


Figure 4.9(a) for RUN-1 (binary system)

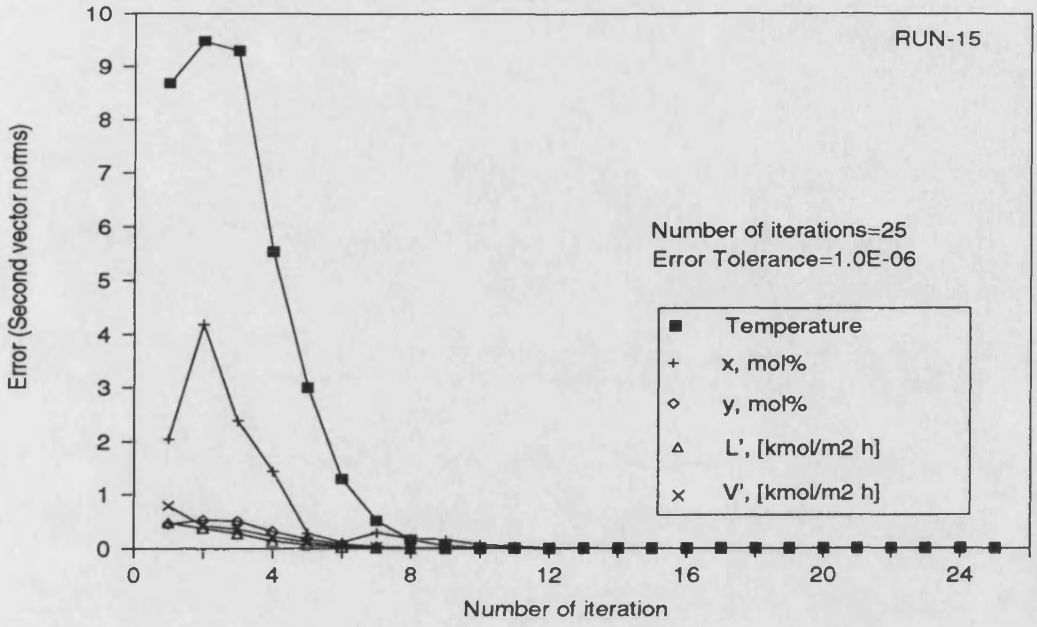


Figure 4.9(b) for RUN-15 (ternary system)

Figure 4.9 Numerical convergency of the total reflux steady state conditions

4.4 Dynamic Product Period

When the reflux ratio is set to a different value from total reflux, the product period starts. This period requires a solution of highly stiff partial and algebraic equation sets. This section covers the behaviour of the solution with different initial conditions and reflux ratio settings. The effect of different boilup rate and reflux ratio operation modes on the solution will be investigated.

Constant Vapour Boilup Rate:

During experiments, heat input to the reboiler controlled manually to keep the pressure drop across the column constant and as a result, vapour boilup rate from the reboiler constant. In order to find out the value of vapour boilup rate for the experiment RUN-15

(ternary system), four runs were carried out by PACBACDIS while keeping the vapour boilup rate constant at following quantities: 0.15, 0.20, 0.25 and 0.35 kmol/hr. Figures from 4.10 to 4.14 show experimental and modelling results of some column variables.

Experimental and model results of temperature profiles of the five different points, z=0 cm, z=42.5 cm, z=85 cm, z=127.5 cm and z=170 cm in the packed section (from the bottom to the top) are compared in Figure 4.10 (a),(b),(c),(d) and (e) respectively.

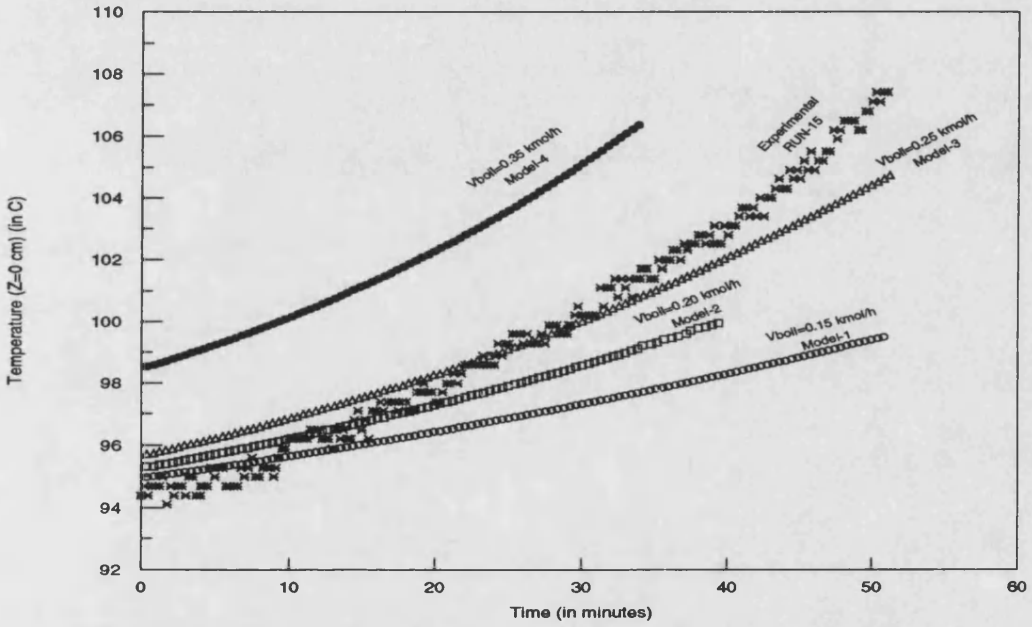


Figure 4.10(a) at the bottom of the packed section (Z=0 cm)

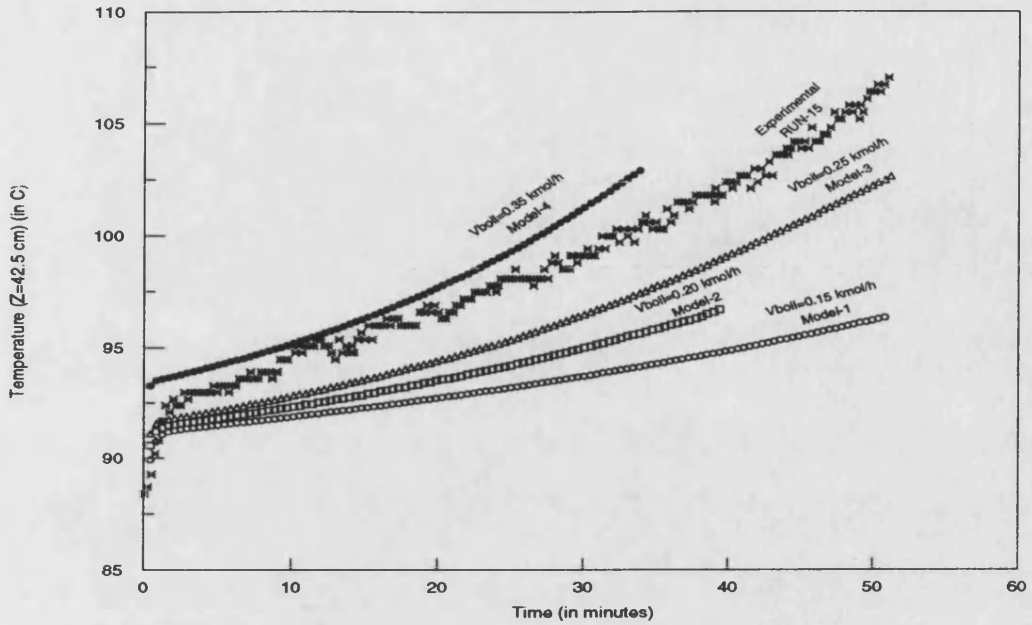


Figure 4.10(b) in the packed section (Z=42.5 m)

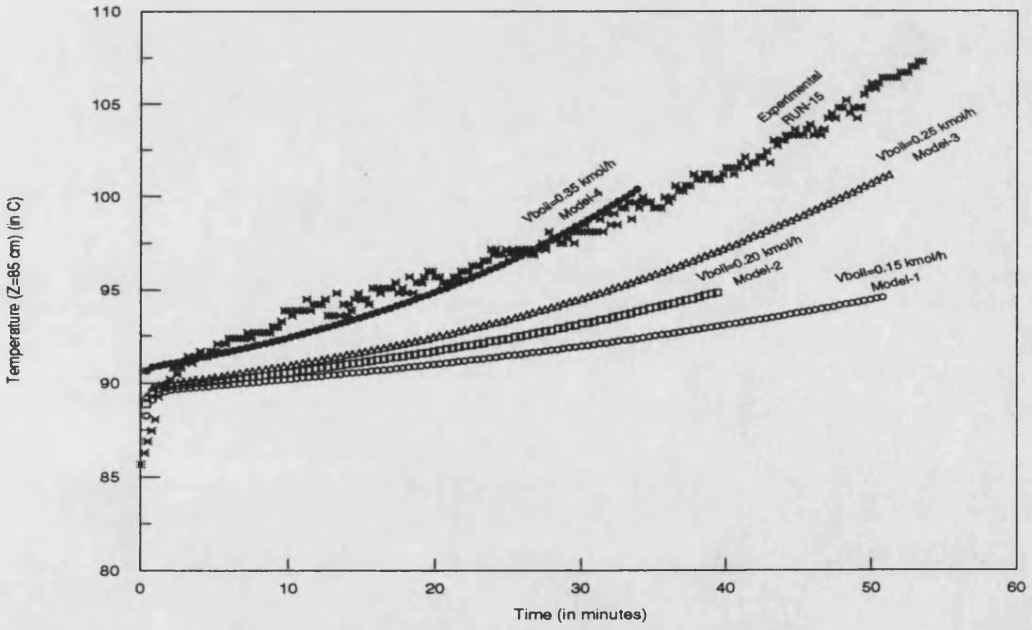


Figure 4.10(c) in the packed section (Z=85 cm)

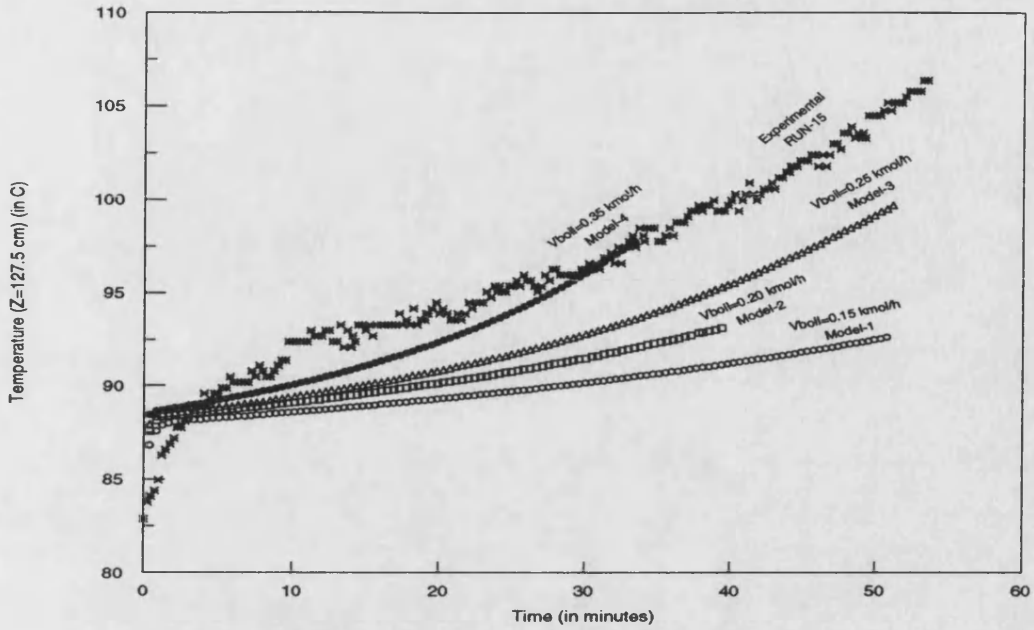


Figure 4.10(d) in the packed section (Z=127.5 cm)

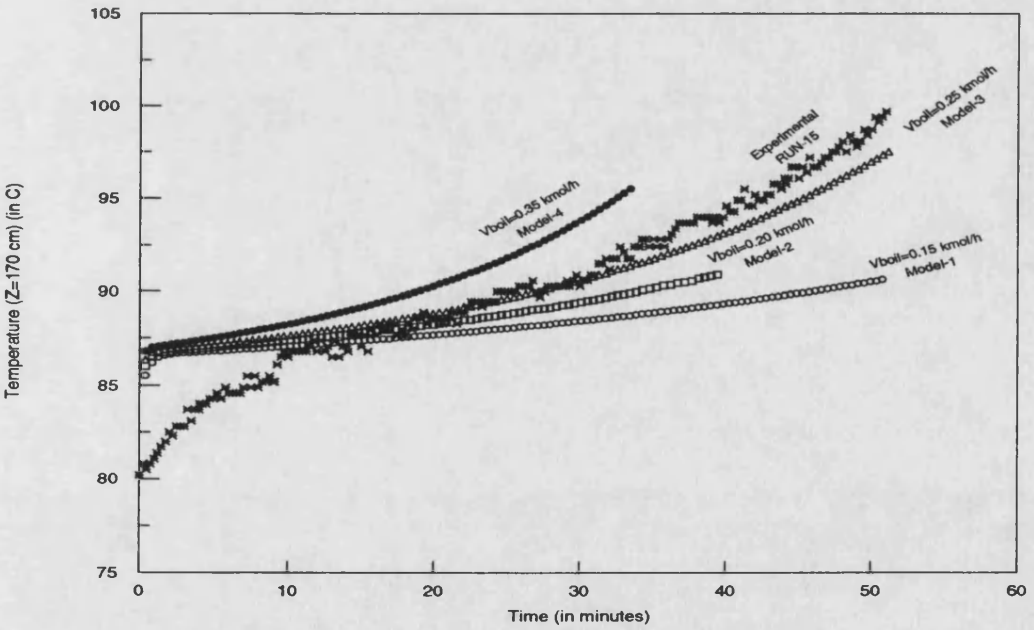


Figure 4.10(e) on the top of the packed section (Z=170 cm)

Figure 4.10 Comparison between experimental and modelling results of temperature profiles with constant vapour boilup rate variations.

Figure 4.11 shows the pressure drop values across the column of the experiment RUN-15 and simulation results of PACBACDIS. The experimental values lie between the simulation results of using vapour boilup rates of 0.20 and 0.25 kmol/hr.

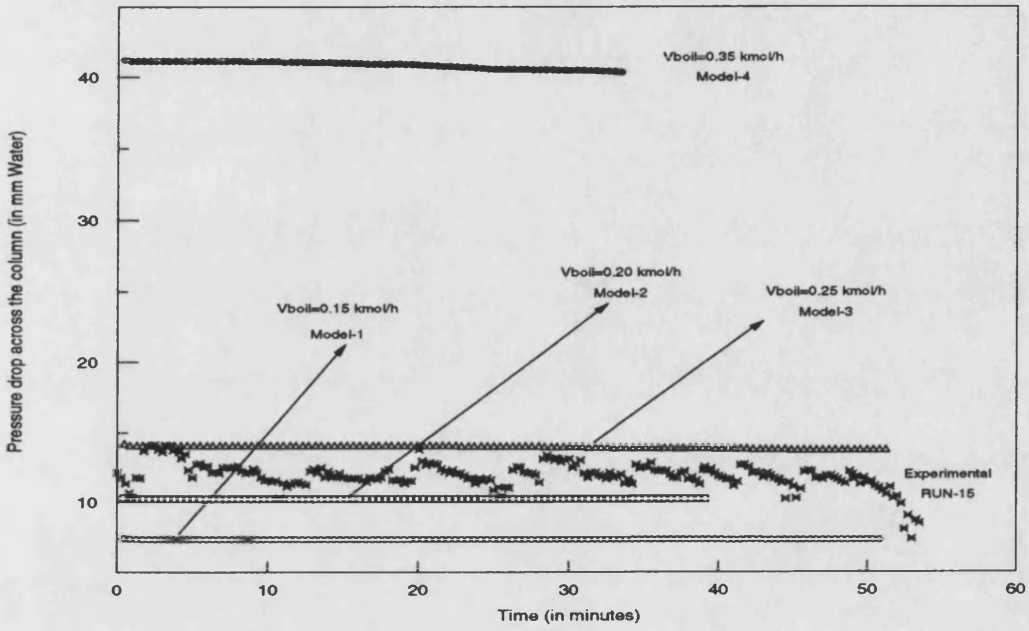


Figure 4.11 Comparison between experimental and modelling results of pressure drop profiles across the column with constant vapour boilup rate variations from the reboiler.

Experimental and simulation results of overhead composition profiles are given in Figure 4.12 for cyclohexane, n-heptane and toluene in (a), (b) and (c) respectively.

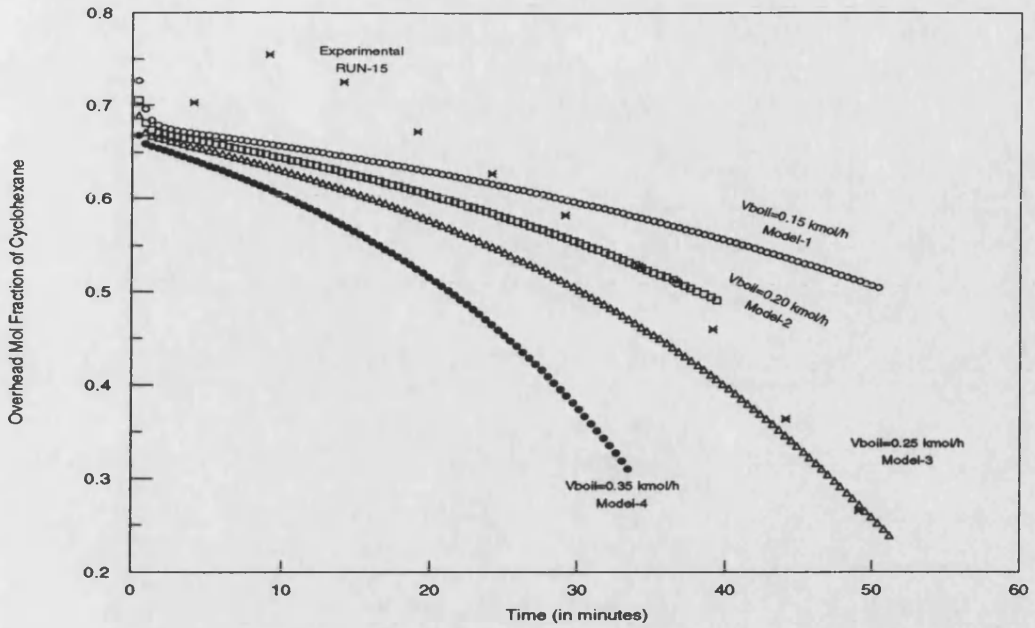


Figure 4.12(a) Mol fraction of Cyclohexane in overhead mixture

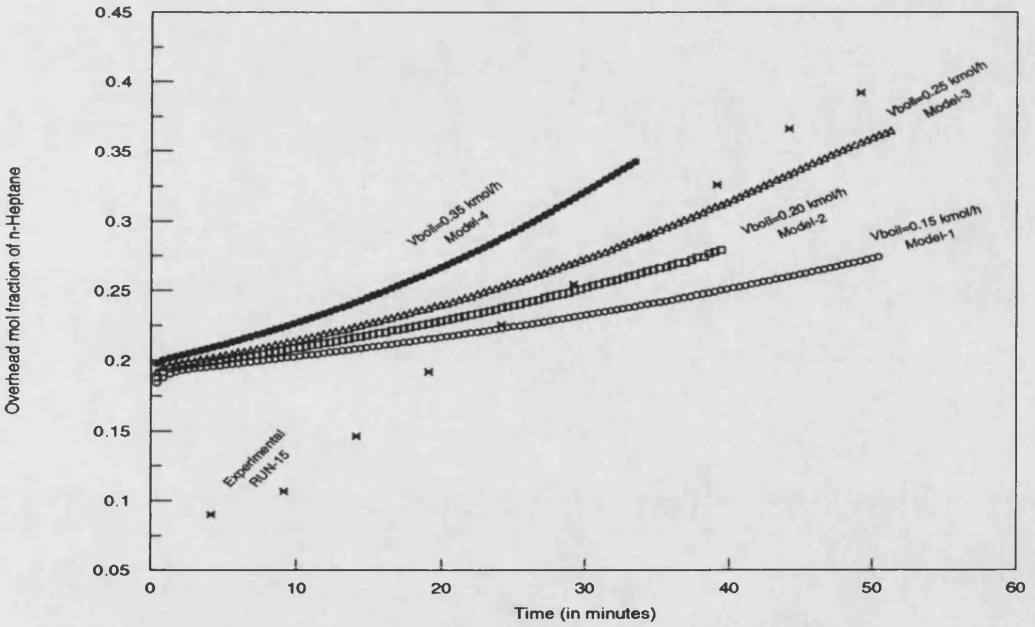


Figure 4.12(b) Mol fraction of n-Heptane in overhead mixture

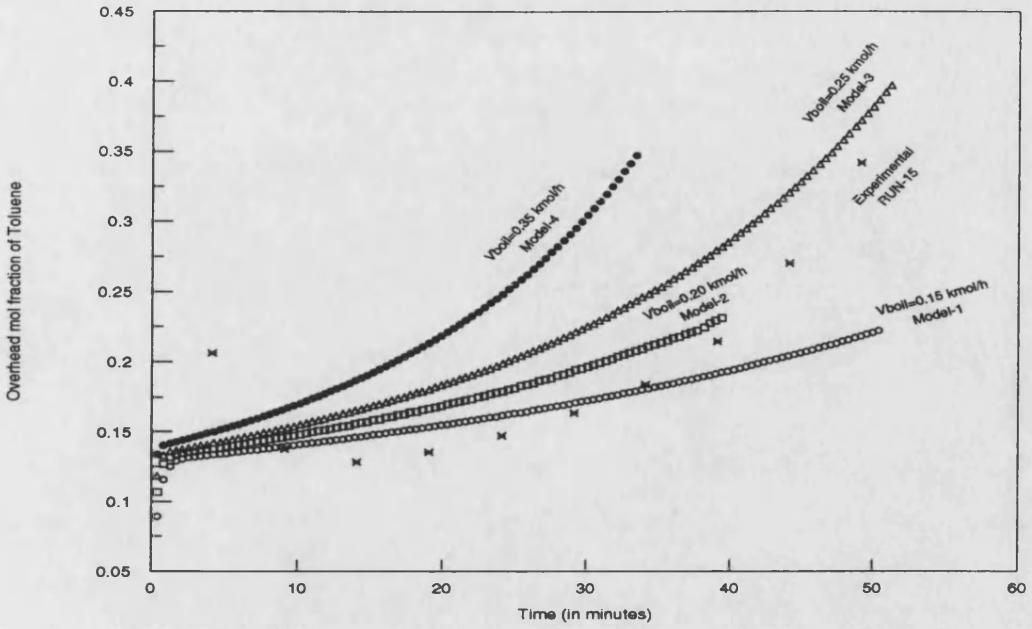


Figure 4.12(c) Mol fraction of Toluene in overhead mixture

Figure 4.12 Comparison between experimental and modelling results of overhead mol fraction profiles with constant vapour boilup rate variations from the reboiler.

Due to the nonavailability of the experimental data for reboiler composition profiles of RUN-15, Figure 4.13 illustrates only the simulation results of vapour boilup rate effects on the composition profiles of reboiler.

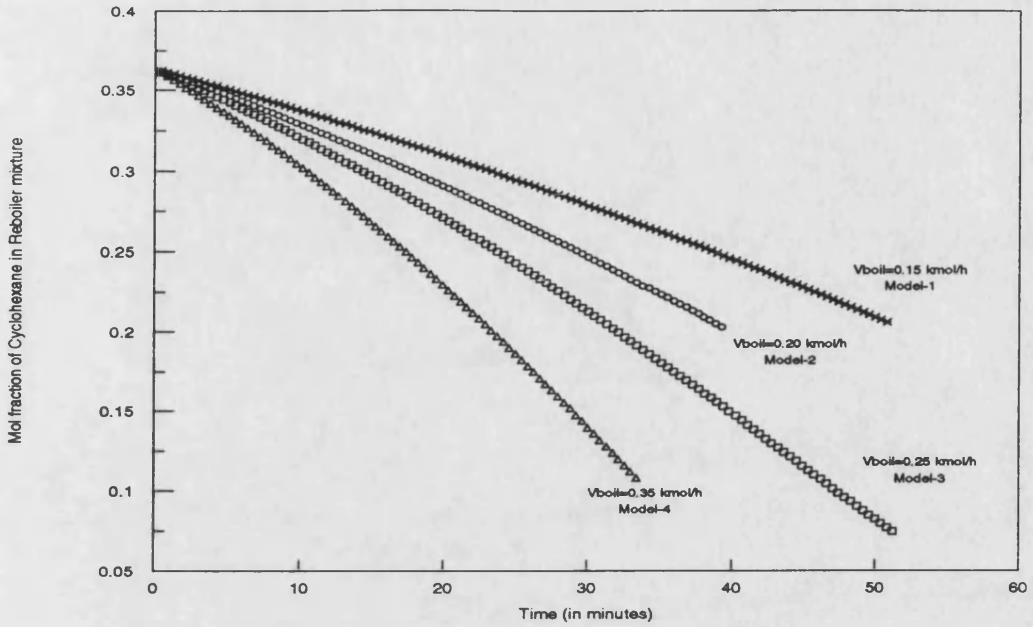


Figure 4.13(a) Cyclohexane in reboiler mixture

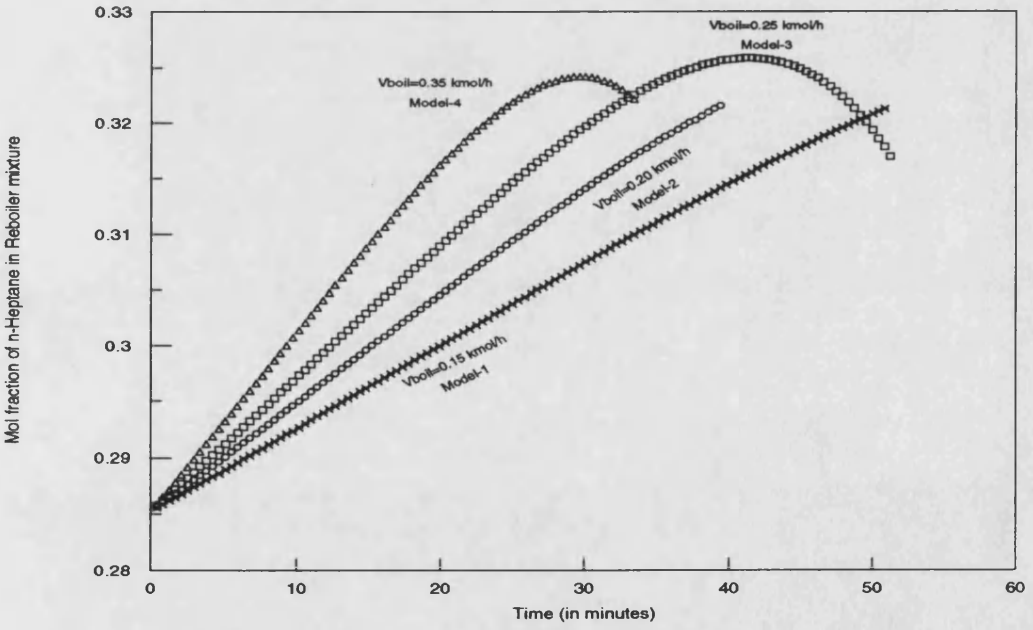


Figure 4.13(b) n-Heptane in reboiler mixture

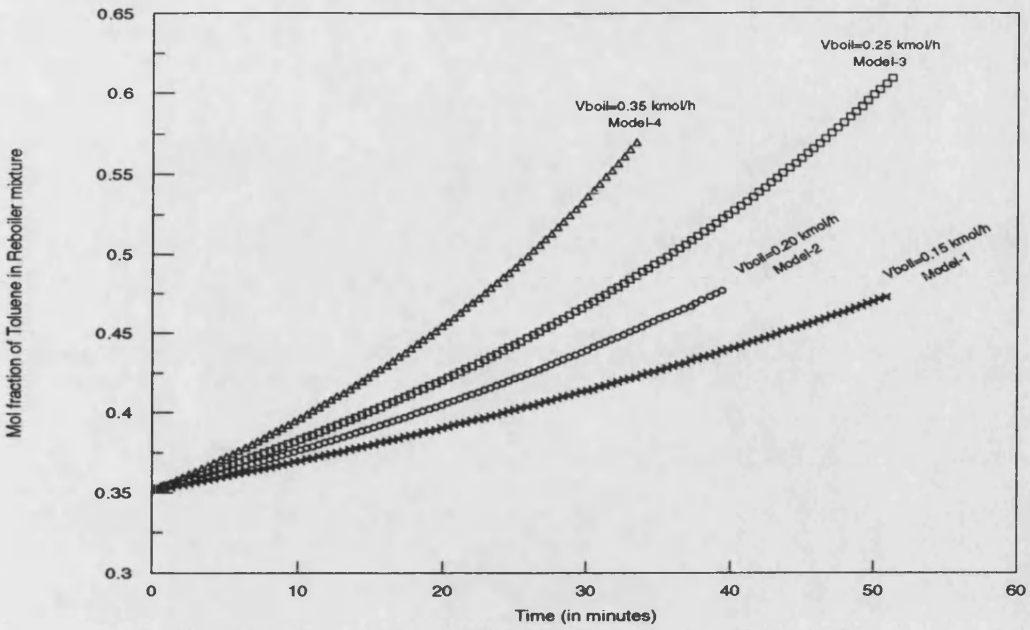


Figure 4.13(c) Toluene in reboiler mixture

Figure 4.13 Modelling results of reboiler mol fraction profiles with constant vapour boilup rate variations from the reboiler.

Reboiler heat input rate, condenser heat output rate and the amount of reboiler charge are also the important variables of the batch distillation system. Figure 4.14 shows their profiles throughout the simulation and/or the experimental runs.

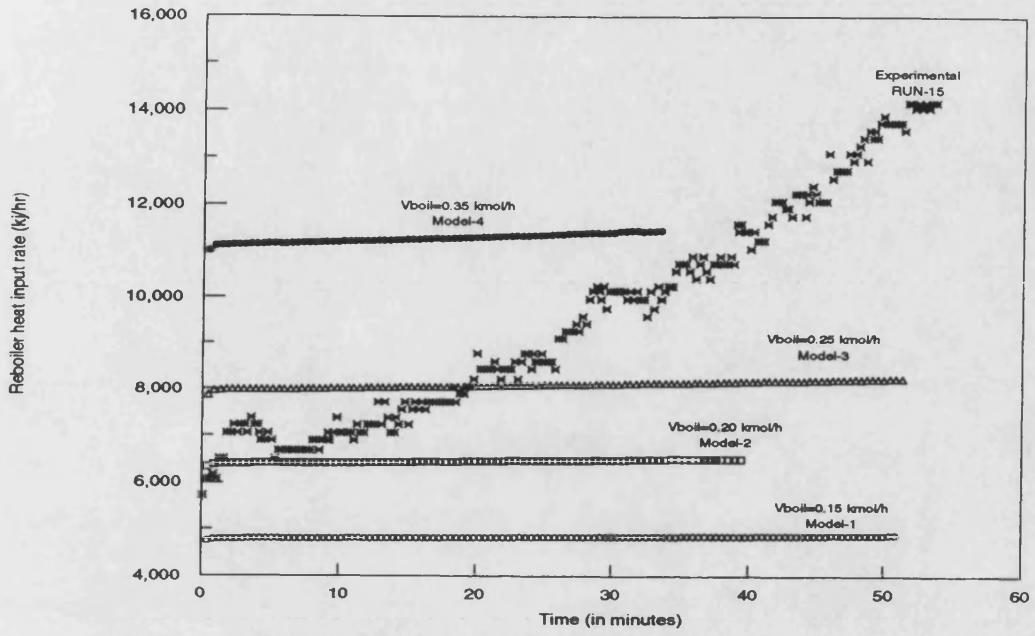


Figure 4.14(a) Heat input rate to the reboiler

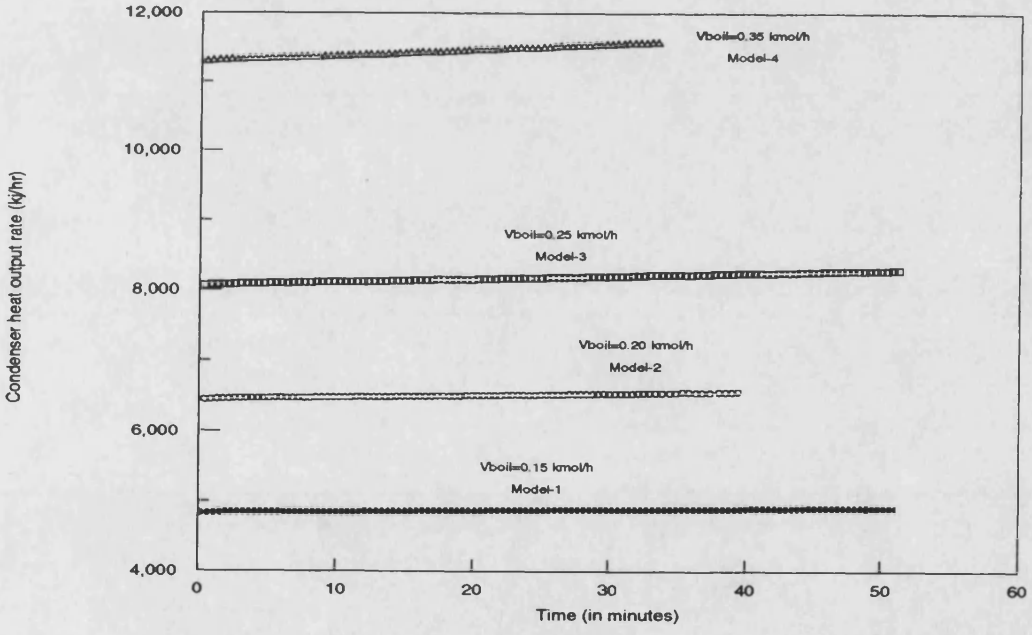


Figure 4.14(b) Heat output rate from the condenser

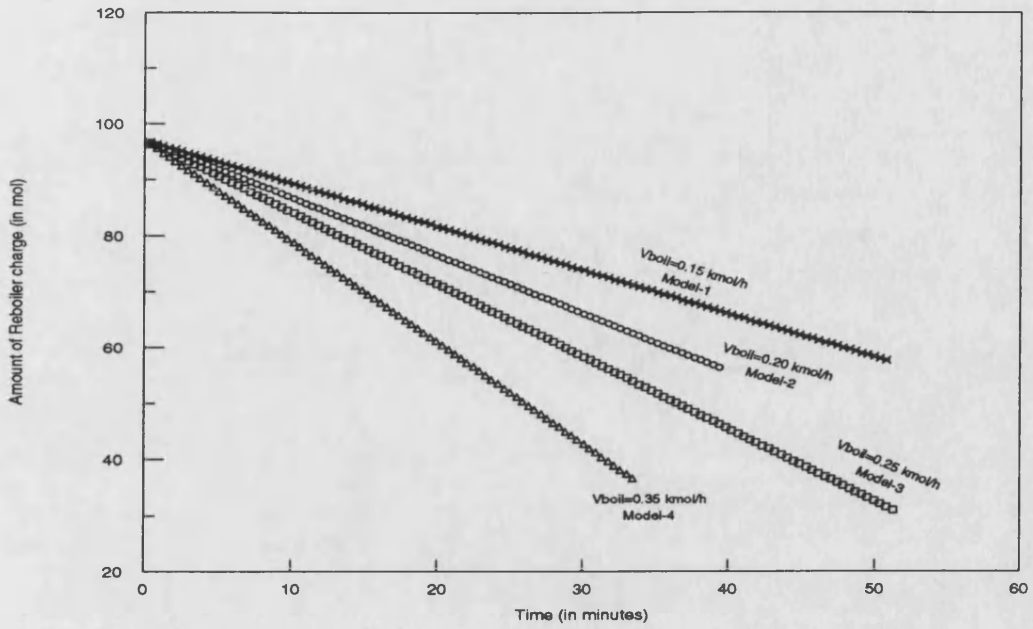


Figure 4.14(c) Amount of the mixture in the reboiler

Figure 4.14 Profiles with constant vapour boilup variations

It is clear from these figures that the assumption of constant boilup rate throughout the experimental column operation is not correct and the boilup rate is changing during the run. This fact was actually observed during experimental runs. The manual control of the boilup rate is not the only reason for a variable boilup rate but there was a upper limit for the steam pressure available on the line. Because of the capacity of the main boiler, the maximum steam pressure was 30 psig. When the composition of heavier components of the charge mixture increased in the reboiler, the available steam heating qualities were not enough to keep the boilup rate constant.

Variable Vapour Boilup Rate

Two more runs were carried out using the same experiment (RUN-15) and input data but varying the boilup rate. Instead of keeping the boilup rate constant, different profiles of heat input rates, Qreb were tested for the simulation runs to predict the boilup rate profiles of the experiment. In the first simulation run, Qreb was kept constant at 6000 kJ/hr (given as Model-5 in Figure 4.15) and in the second simulation run, it was varied as a function of time (Model-6, obtained from curve fitting to the experimentally recorded data). Figure 4.15 shows that Model-6, the variable boilup rate with variable heat input as a function of time gives the best fitting with the experimental data. Therefore, it can be concluded that the vapour boilup rate changes according to changing heat input rate to the reboiler as a function of time.

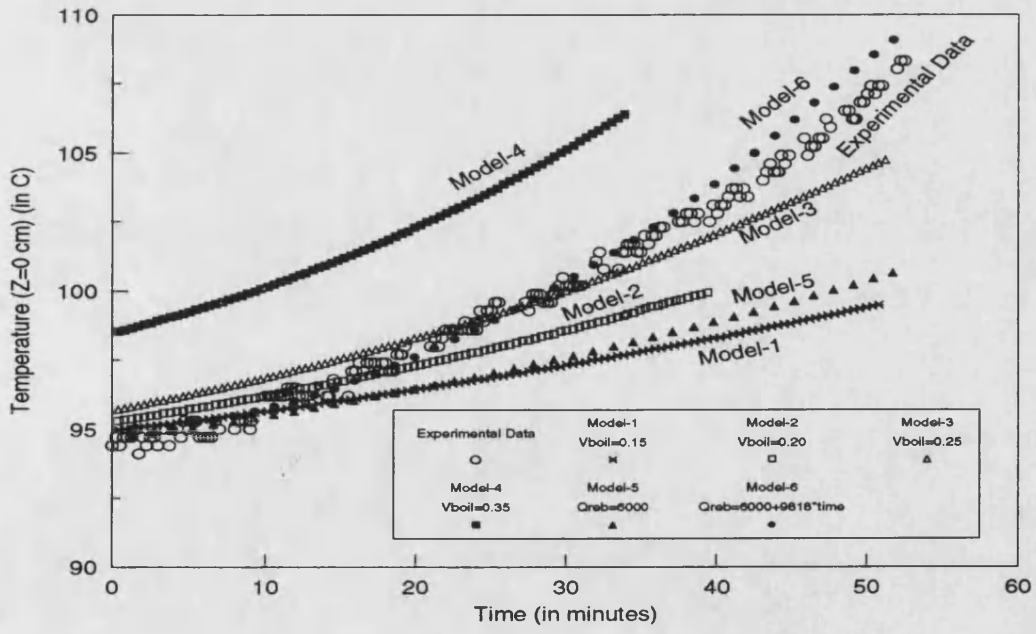


Figure 4.15 Effect of different boilup rate operation modes on the separation results for RUN-15.

In order to see the effect of reboiler heat input profiles on the operation, another simulation run by PACBACDIS was made with a different time function of Qreb. Figure 4.16 illustrates that the difference is significant between two model results and accurate Qreb profiles should be predicted for the experimental runs for the sake of accuracy of PACBACDIS predictions of the column variables, such as temperatures and compositions.

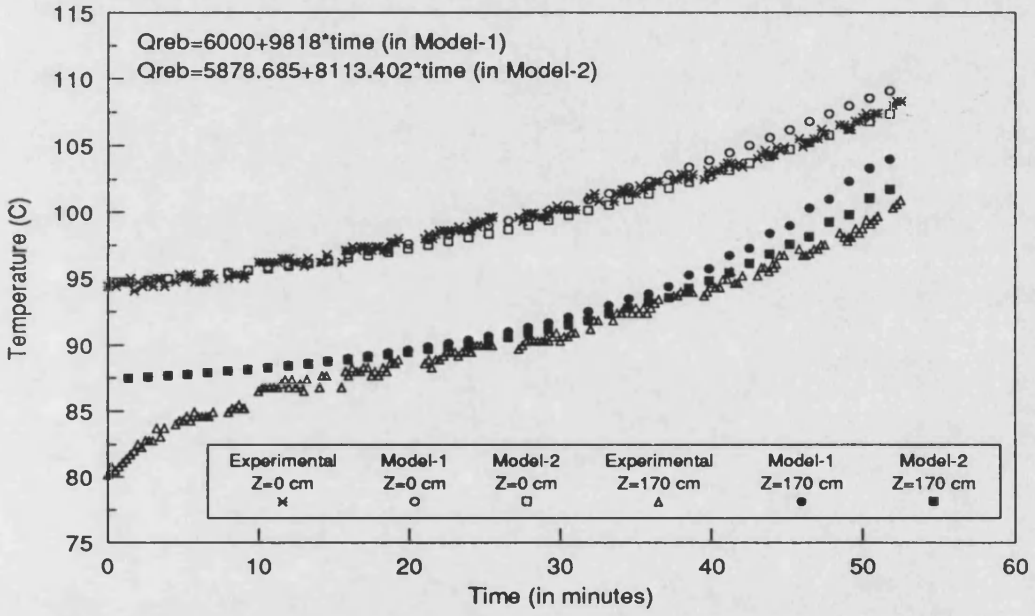


Figure 4.16 Effect of different heat input profiles on the column temperature profiles.

As stated earlier, measured steam temperatures and steam mass flow rates were recorded during experiments. Heat input to the reboiler can be calculated by regression of these experimental data and the data from the steam tables. The equations derived are as follows:

Latent heat of steam [kJ/kg]:

$$\lambda = 2535.523 - 2.77035 * Temp_{steam} \quad (4.1)$$

Where temperature of steam is in [C] and the regression coefficient is 0.9999.

Steam pressure [psig]:

$$P = 63.53 - 1.74575 * Temp_{steam} + 0.011117 * Temp_{steam}^2 \quad (4.2)$$

The regression coefficient is 0.9999.

Steam mass flow rate [kg/h]:

$$m = -27.904423 + 0.255269 * Temp_{steam} \quad (4.3)$$

The regression coefficient is 0.9999.

Heat input to the reboiler [kJ/hr]:

$$Qreb = m * \lambda \quad (4.4)$$

Experimental heat input rates were calculated by the equations from (4.1) to (4.4) and results were included in experimental data files (names listed in Table 3.6). Later, these heat input rate calculations were defined as functions of time for simulation purposes. Figure 4.17 shows the heat input profile for RUN-15. Coefficients of the regression equations, a and b of [Qreb=a+b*time, time is in hr] and the maximum Qreb values of the experiments allowed by the main boiler and the regression coefficients are listed in Table 4.2.

Table 4.2 Regression Data for Heat Input Profiles of Experiments

Experiment	a	b	Maximum Qreb	Regression Coefficient
RUN-1	6456.21	10489.98	13102	0.84
	7800.0 for 0≤time<0.4 13102.0 for 0.4<time≤0.8	0	-	-
RUN-2	6577.19	10769.08	13102	0.92
RUN-3	6757.23	12887.72	13102	0.97
RUN-4	9151.75	20531.74	13102	0.93
RUN-5	7997.71	10085.11	13425	0.96
RUN-6	9179.37	12883.52	12565	0.92
RUN-7	12565	0	12565	-
RUN-8	10609.34	17484.97	12565	0.70
RUN-9	9313.33	16748.08	12565	0.75
RUN-10	8732.91	5279.12	12940	0.93
RUN-11	7050.52	12930.18	12730	0.94
RUN-12	7531.69	11016.90	13200	0.98
RUN-13	6273.97	14936.29	13350	0.98
RUN-14	7914.45	12934.35	13350	0.96
RUN-15	5560.92	8997.84	13585	0.96
RUN-16	5267.98	3471.12	-	0.95
RUN-17	4466.46	3330.87	-	0.98
RUN-18	5123.63	2000.69	-	0.98
RUN-19	4181.49	4976.34	-	0.96

Unless it is stated, later runs will consider a variable boilup rate with a variable heat input rate as a function of time which coefficients were given in Table 4.2.

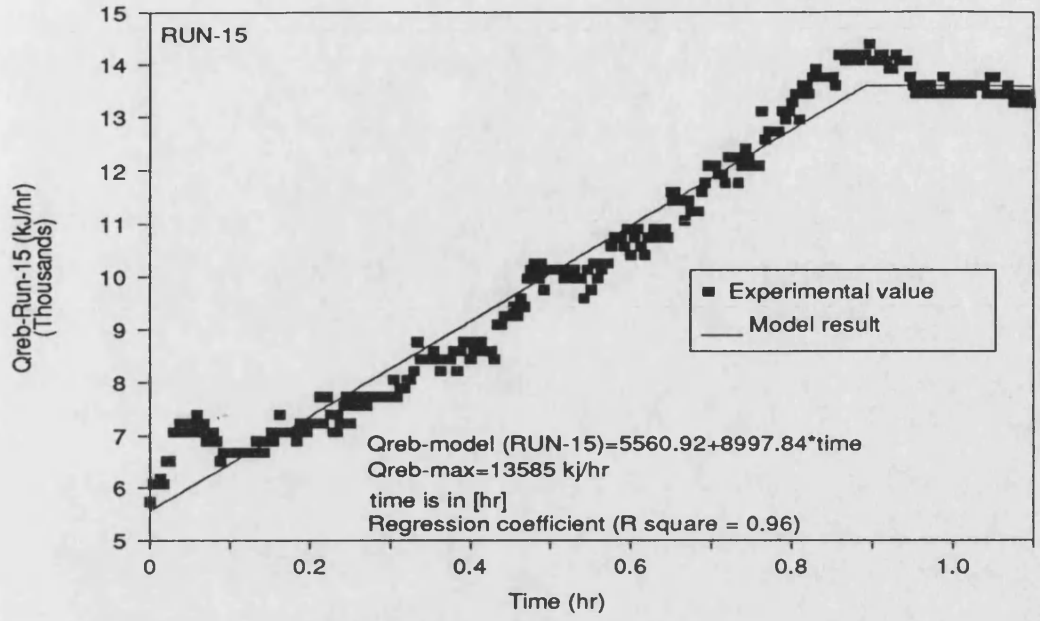


Figure 4.17 Experimental data and regression results for heat input profile of RUN-15.

Column Pressure

Recalling from Chapter 2, PACBACDIS solves a model that considers column pressure effects on the separation variables. Experimental data for the column top and bottom pressures are given in Table 4.3. These pressure values were averaged over the experiment time period.

A different experiment, RUN-18, data were used in order to study the effect of variations in the average top column pressure on the simulation results. The measured average top column pressure (0.9705 atm) was increased by 0.1% to 0.9715, 1% to 0.9805 and 10% to 1.0705 and these values were used as input data into the simulation program. Figure 4.18 shows the temperature results as an example for the effect of these pressure values. It can be seen that 1% error made in top column pressure measurements results in a

noticeably higher yet acceptable profile. However a 10% error shifts the temperature profile considerably. This shows that an experimental error over 1% in top column measurements could cause unacceptable simulation predictions.

Table 4.3 Experimental Data for Average Column Pressures

Experiment	Top Column Pressure [atm]	Bottom Column Pressure [atm]
RUN-1	0.9735	0.9750
RUN-2	0.9795	0.9805
RUN-3	0.9855	0.9865
RUN-4	0.9900	0.9915
RUN-5	0.9899	0.9910
RUN-6	0.9835	0.9845
RUN-7	0.983	0.9840
RUN-8	0.9928	0.9939
RUN-9	0.9909	0.9917
RUN-10	0.9899	0.9910
RUN-11	0.9899	0.9910
RUN-12	0.9810	0.9821
RUN-13	0.9849	0.9862
RUN-14	0.9840	0.9851
RUN-15	0.9790	0.9800
RUN-16	0.9660	0.9685
RUN-17	0.9695	0.9707
RUN-18	0.9705	0.9720
RUN-19	0.9830	0.9841

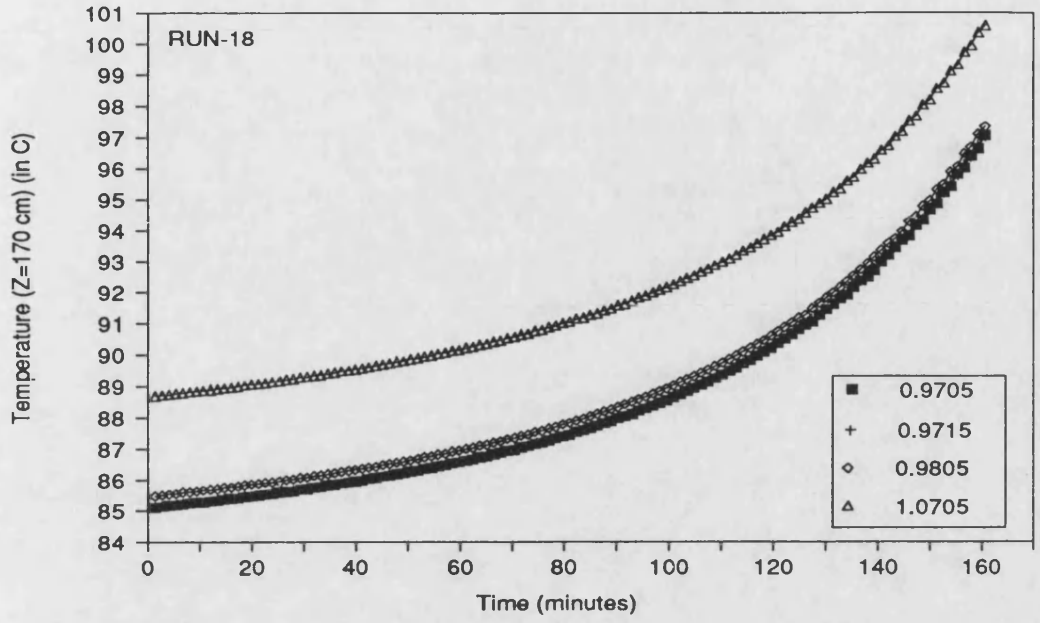
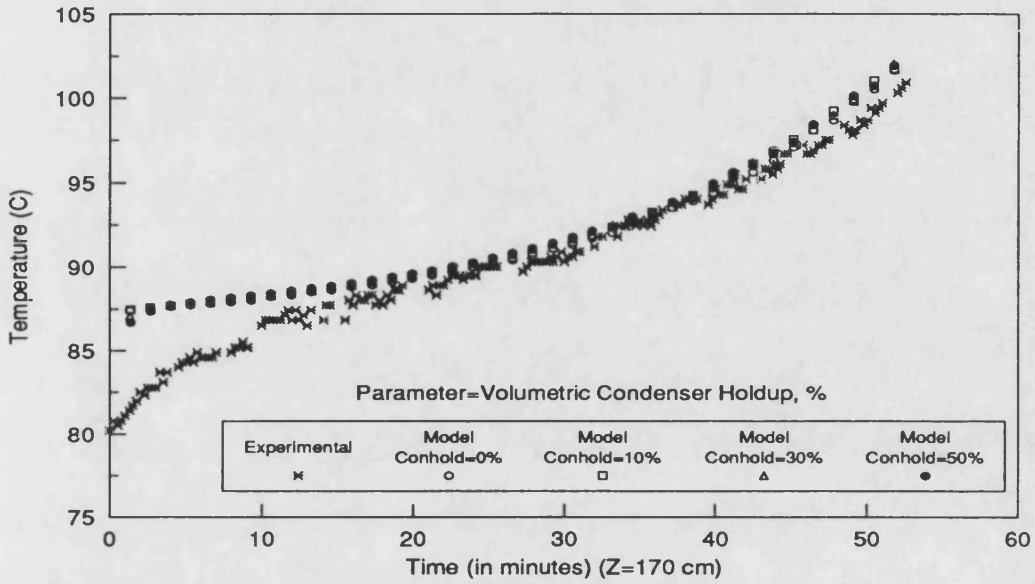


Figure 4.18 Effect of top column pressure on column temperature profile.

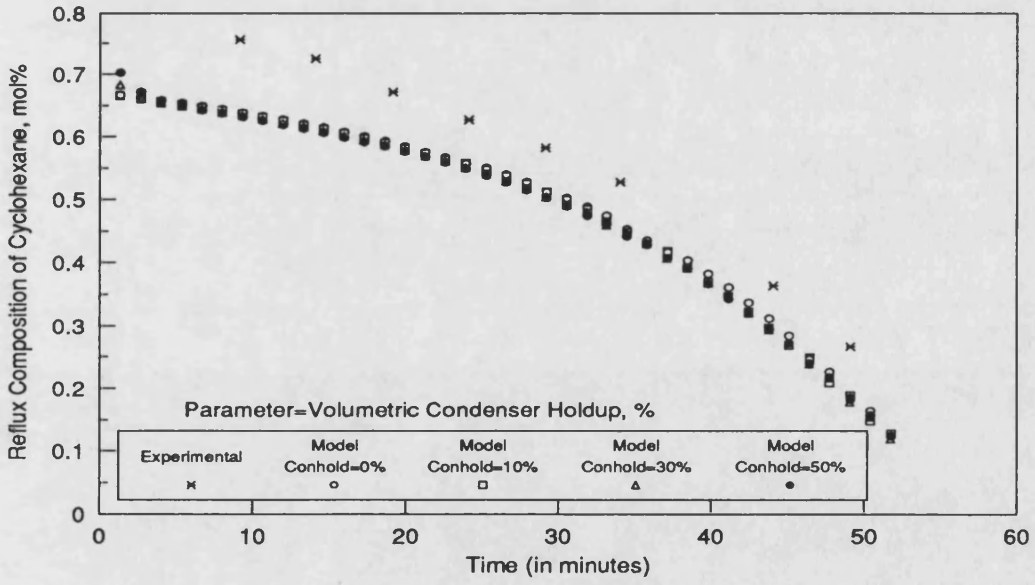
Condenser Holdup

The model considers the condenser holdup. In order to find out its effect, a number of computer runs were carried out with 0%, 10%, 30% and 50% volumetric condenser holdups. Figure 4.19 shows that the effect is not very significant for the experimental column due to the fact that the reflux drum and the condenser volumes are relatively small compared to the column and reboiler volumes. For the column in use a 10% volumetric condenser holdup makes 0.566 mol liquid holdup when the reboiler liquid holdup is 95.8 mol. This means that the condenser holds 0.59% molar holdup of the total mixture charge. However, the condenser holdup will have a greater effect on separation for bigger size industrial columns. The increasing condenser holdup will increase the concentration of the more volatile component in the distillate at any given time. The reason for this is that accumulation of liquid in the condenser receiving vessel effectively

averages the composition of the vapour currently leaving the column with the compositions of all the overhead vapour which preceded it. This will cause a delaying effect on the final removal of higher volatility compounds from the column.



(a) Temperature profile



(b) Composition profile

Figure 4.19 Effect of condenser holdup variations using the experimental data of RUN-15

Liquid Holdup in the Column

During the parametric study, the effect of liquid holdup in the column on steady state total reflux results was investigated. It was observed that the total reflux steady state compositions are a function of holdup, so that all of the curves do not start at the same values of composition. Figure 4.20 verifies this result. The curves with the greater holdup start at slightly lower compositions of the most volatile component (clearer in Figure 4.20 (b)).

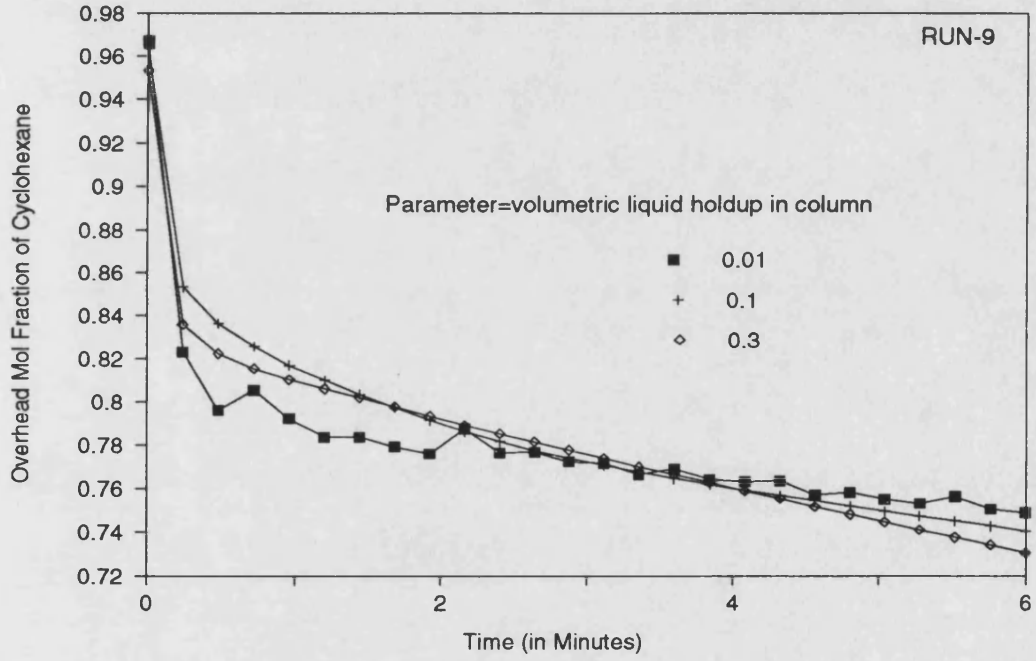


Figure 4.20(a) Overhead composition profile

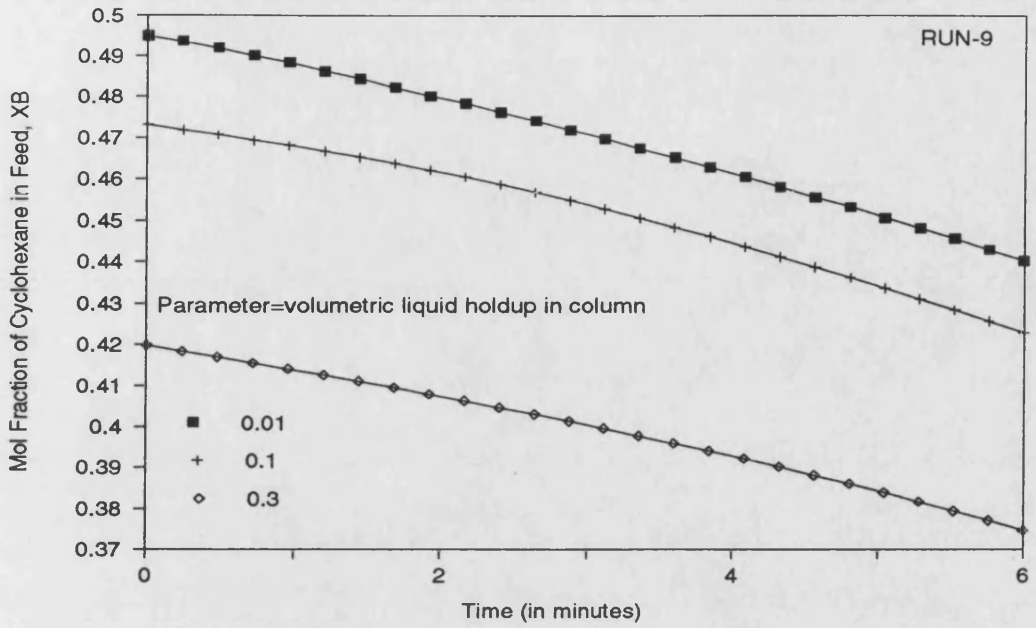


Figure 4.20(b) Reboiler composition profile

Figure 4.20 Effect of liquid holdup in the column using the experimental data of RUN-9

Reflux Ratio

PACBACDIS was tested with three different modes of operation. In the first operation mode case, a constant reflux ratio was defined for the simulation during a product period. Initial conditions of RUN-18 were given to the computer runs. Runs were repeated with the same initial conditions but the constant reflux ratio (=L/V) was defined as 95%, 90%, 80% and 70%. Figure 4.21 shows the effect of reflux ratios on overhead composition profiles. Thus, as expected, the higher the reflux ratio, the higher the maximum obtainable Cyclohexane purity.

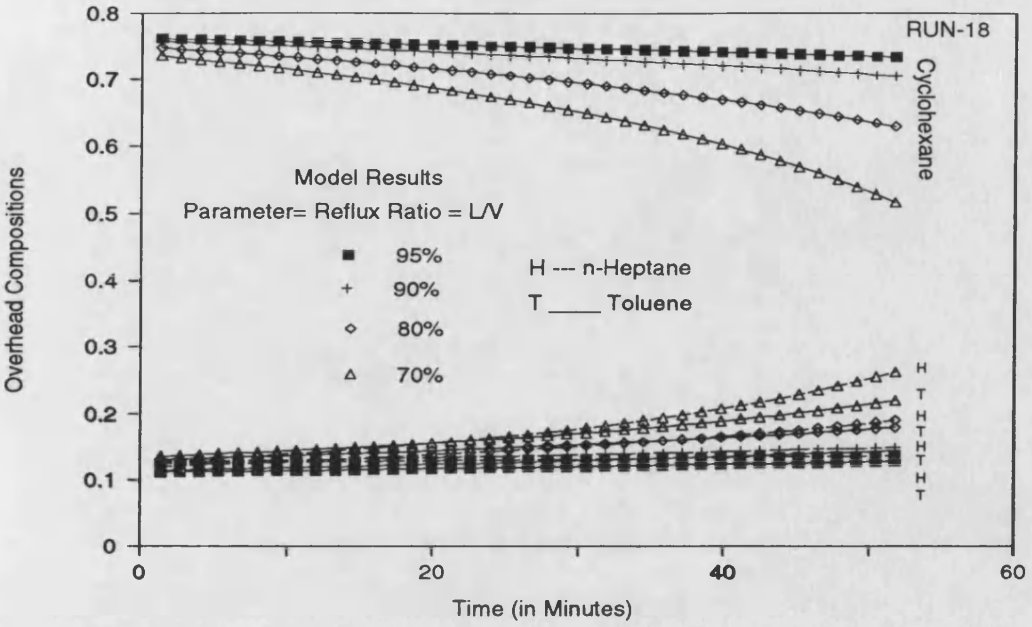


Figure 4.21 Effect of constant reflux ratio on overhead concentration profiles

In the second operation mode, the reflux ratio was varied stepwise. Figure 4.22 illustrates the experimental and calculated Cyclohexane overhead composition profiles for RUN-11.

The experimental composition profile of a stepwise variable reflux ratio operation mode did not match the calculated composition profile sufficiently. The reason for this disagreement could be explained by the fact that the most important variable among input variables of the simulation is the reflux ratio and the exact switching times. These variables are nevertheless experimentally very difficult to measure. Nad and Spiegel (1987) stated that measurement errors could be in the range of 10%, at least. The best fit could be obtained by trying different reflux ratio and the switching times.

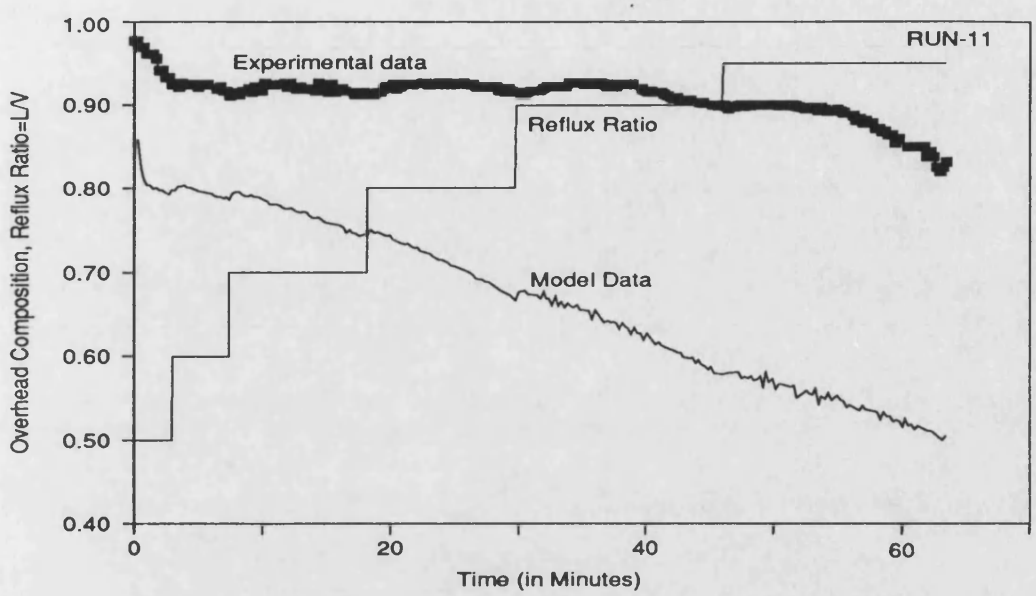


Figure 4.22 Effect of step changes in reflux ratio on overhead concentration

In the last operation mode, the reflux ratio was controlled to keep the overhead Cyclohexane concentration at its set point 95%. It could be seen from Figure 4.23 that, the agreement between the experimental data and the model results is very good, bearing in mind the multiparametric nature of the packed batch distillation operation.

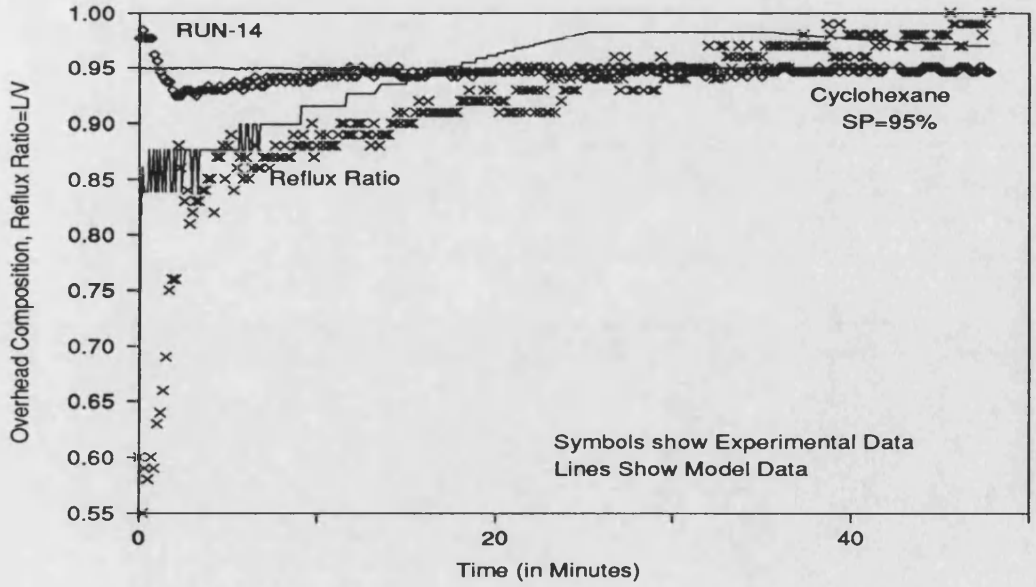


Figure 4.23 Controlled reflux ratio

DASSL versus Finite Difference

As stated earlier in Chapters 1 and 2, two methods were used for solving partial differential-algebraic equation sets, DASSL and the Finite Difference. The nature of Newton's method used in DASSL made it necessary to use a second approach in case of a failure. In other words, if DASSL fails to converge with a negative IDID, PACBACDIS will continue the simulation using Finite difference routines. Figures 4.24 and 4.25 show that the difference between results of these two methods are not significant but DASSL converges with a less number of corrector iteration steps.

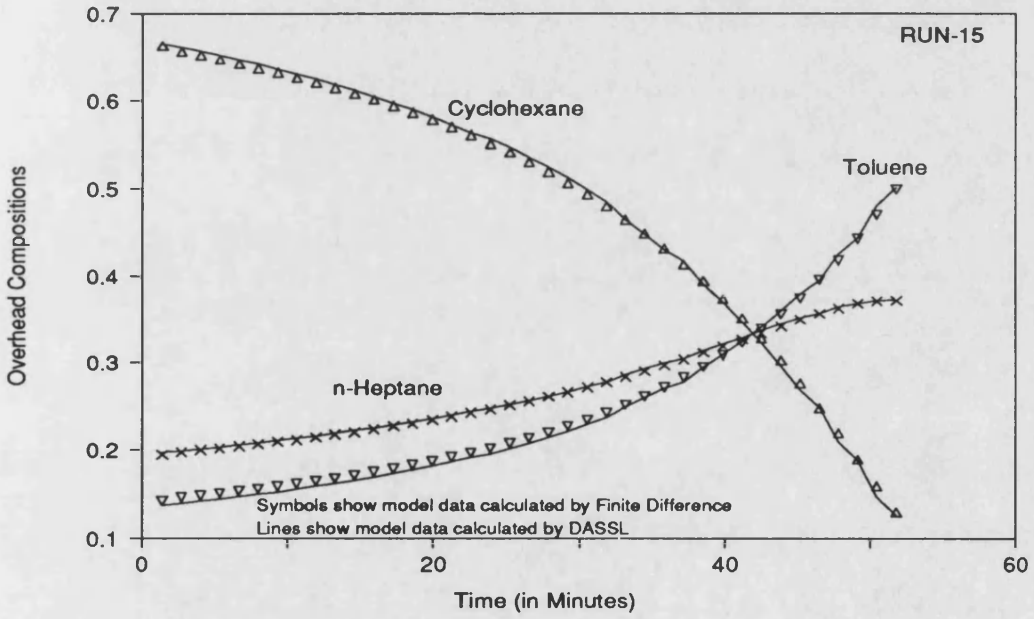
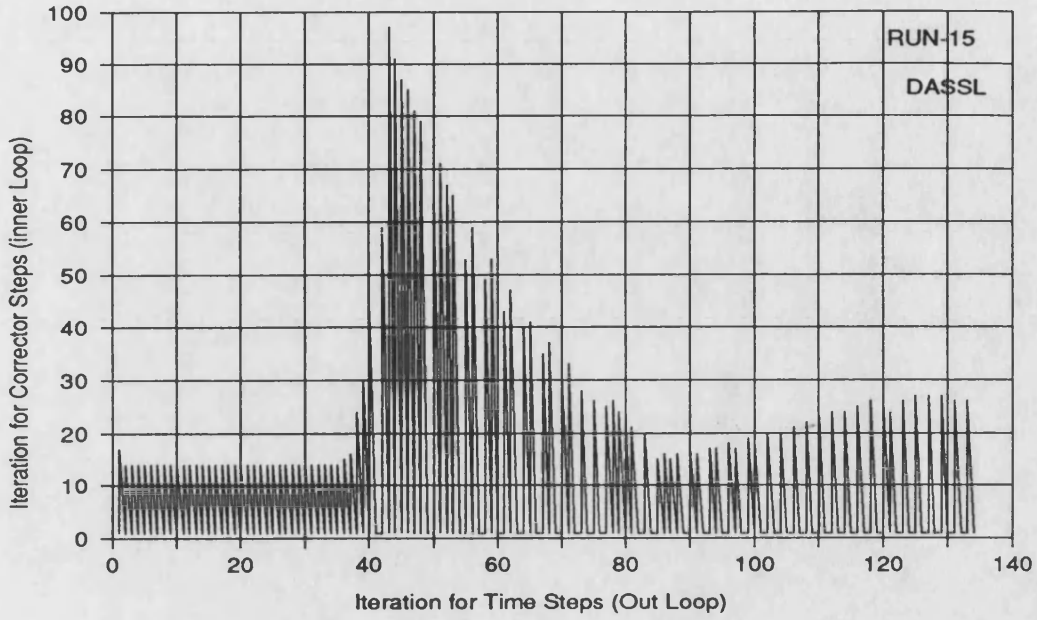
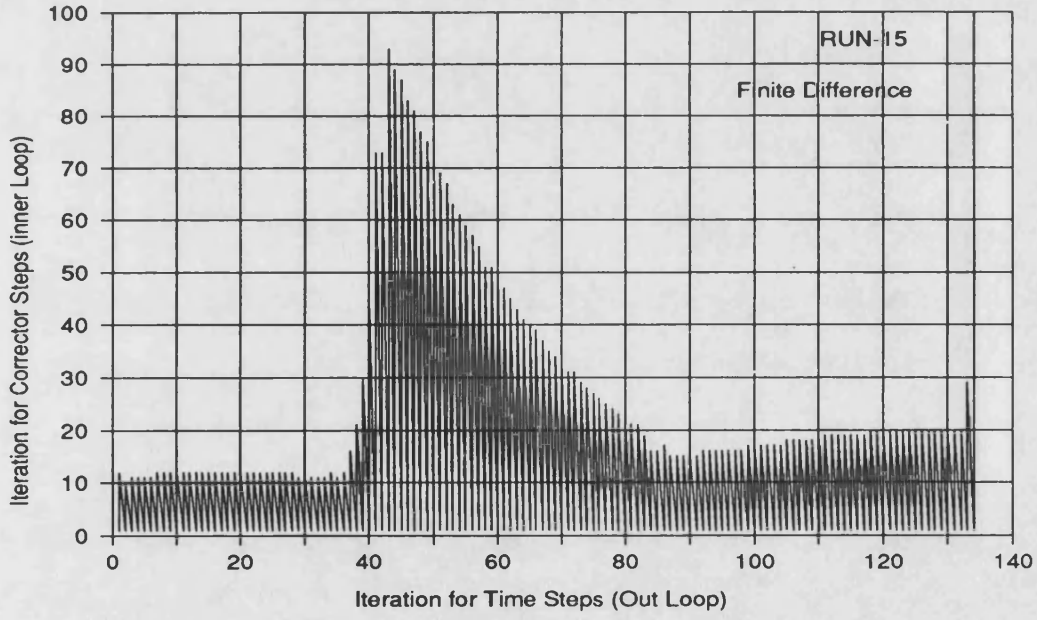


Figure 4.24 Concentration profile results of DASSL and Finite Difference methods



(a) using DASSL method



(b) using Finite Difference method

Figure 4.25 Convergence profiles

4.5 Conclusion

Application of these operational test conditions demonstrates that predictions of the model compare well with experimental results. These tests include the behaviour of the column under steady state and dynamic conditions and therefore the model can be said to simulate the whole nature of the system. The program results can easily be improved by defining the system parameters by their correct values.

SUN SparcStation 4/60C was used as the computing platform. Cumulative process CPU time was taken as the run time and the time changes significantly with operation modes and values of the column and simulation parameters. For example, RUN-11 with stepwise variable reflux took 54.15 hours for 1057 time iteration steps.

PACBACDIS has been designed to be a comprehensive simulation program which can handle a variety of problems in multicomponent, packed batch distillation. The program has a great potential because of its inherent flexibility to all possible modes of operation.

CHAPTER 5
OPERATION MODES FOR MULTICOMPONENT BATCH DISTILLATION COLUMNS

In multipurpose, flexible plants, batch distillation constitutes an efficient separation tool consisting of a series of operation steps with different goals and specifications on product purities. This separation technique is used when a component is to be extracted from variable or complex charges, or if an impurity accumulation may occur during a continuous distillation task. A single column can also treat a wide range of feed compositions, a number of components, and degrees of difficulty of separation. The flexibility of this separation method makes it possible to adjust the operating conditions to the required purity specifications. The operation can be carried out with constant reflux ratio, with constant overhead compositions, or with an optimal policy suitable for the problem under consideration. The choice of operation mode depends upon characteristics of the specific system, the product specifications, and the engineer's preference in setting up a control sequence.

This chapter is planned to cover all operation modes of batch distillation that have been introduced in the literature. The operation modes are summarized under five sections. The first section is allocated for constant reflux, varying overhead composition operation mode. The second section explores the constant overhead composition, varying reflux operation mode. The experimental and modeling works were carried out for these two operation modes (see Chapters 3 and 4). Example runs are introduced to explain their separation properties. In the third section, the cycling operation mode is given. Although this mode has not attracted much attention in the literature, it was thought that it should be included for the sake of completeness.

Optimum operation and recycle waste-cut operation modes are currently two very popular areas of interest. Extensive literature on these modes is summarized in sections four and five.

5.1 Constant reflux, varying overhead composition

Reflux is set at a predetermined value which is maintained throughout the run. Since liquid composition in the reboiler is changing, the instantaneous composition of the distillate also changes.

Instantaneous distillate composition for a multicomponent batch distillation varies with time as a function of volatility, reflux ratio, and the number of theoretical plates or the height of the packed section. Distillation is continued until the average distillate composition is at the desired value. In the case of a binary system, the overhead is then diverted to another receiver, and an intermediate cut is withdrawn until the remaining pot mixture meets the required specifications. The intermediate cut is usually added to the next batch (recycled waste-cut). For a multicomponent mixture, two or more intermediate cuts may be taken between product cuts.

If the value of reflux ratio, R is chosen correctly, in the case of a binary mixture, the composition of the residue reaches the specified value at the same time as the accumulated distillate attains the required composition. Experimental results, obtained in the present study, can be used to explain the separation profiles of constant reflux operation modes. Three different constant reflux ratios 50%, 70% and 90% were set for batch distillation experiments using a binary system with similar experimental conditions (see Table 3.5 for experimental conditions of RUN-9, RUN-3 and RUN-10 respectively). Figures 5.1, 5.2 and 5.3 illustrate the product periods and experimental separation results of these

constant reflux ratio operation modes. Data for the amount of top and bottom products (D and B) on-specification and cut-point product off-specification (S) and product purities are shown in Figures with elapsed times. It can be concluded from these experimental data that, when the reflux ratio increases, the amount of product on-specification increases while the amount of off-specification cut-point decreases. However, operation times also increase.

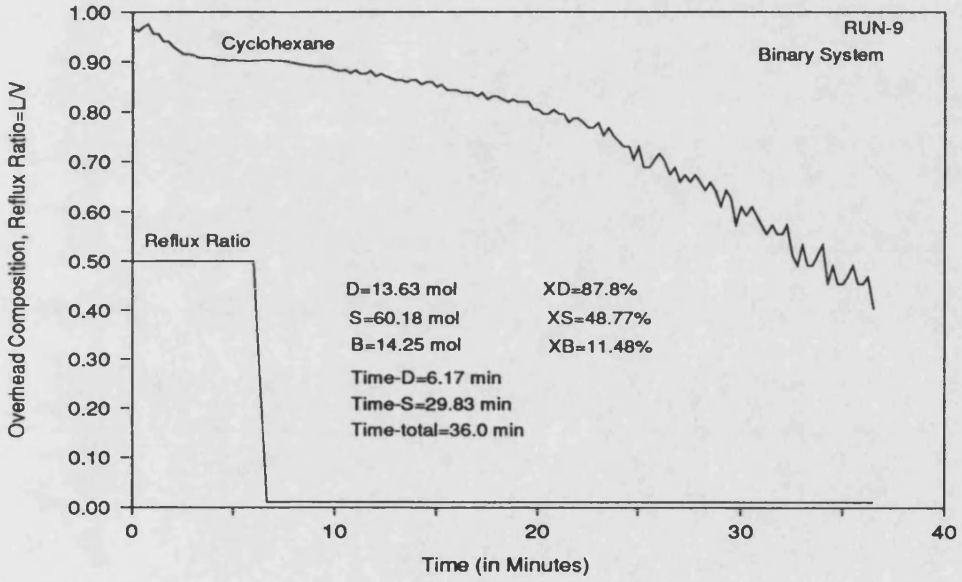


Figure 5.1 Separation profile when Reflux ratio=L/V=50% for the first product period

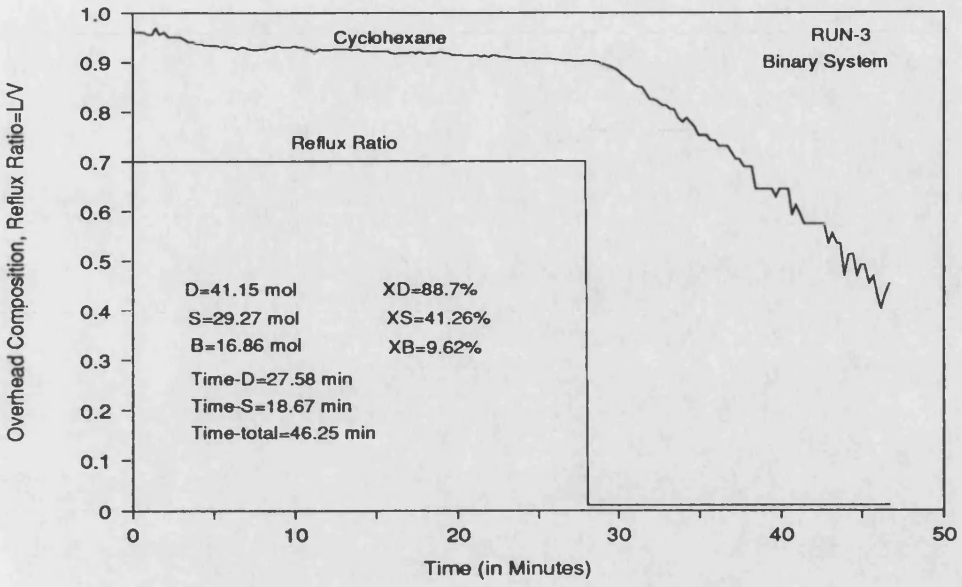


Figure 5.2 Separation profile when Reflux ratio=L/V=70% for the first product period

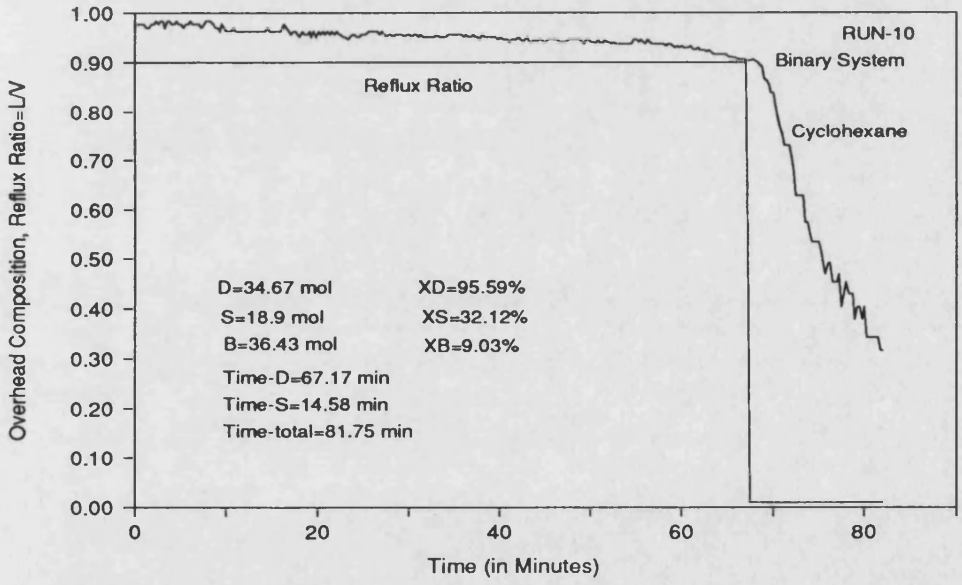


Figure 5.3 Separation profile when Reflux ratio=L/V=90% for the first product period

Piece-wise constant reflux operation mode is another application used for batch distillation columns. Reflux ratio is increased stepwise during the operation. RUN-11

from current experimental work is an example experiment for this operation. As can be seen from Figure 5.4, piece-wise constant reflux operation mode gives the smallest cut-point product amount while saving time compared to 90% constant reflux ratio operation.

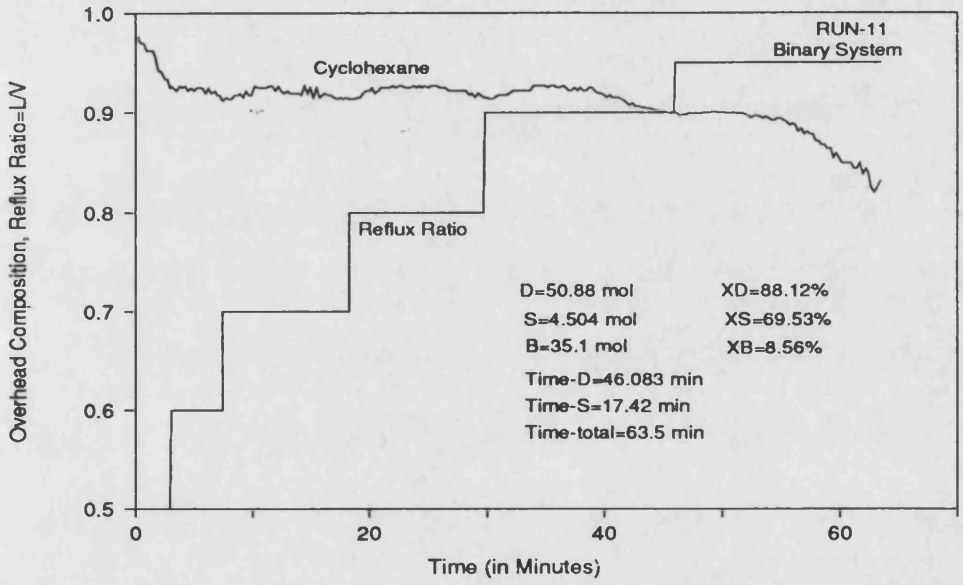


Figure 5.4 Separation profile for the constant reflux step changes

5.2 Constant overhead composition, varying reflux

Most commercial batch distillations are designed for variable reflux control in order to maintain a constant overhead composition.

If it is desired to maintain a constant overhead composition in the case of a binary mixture, the amount of reflux returned to the column must be constantly increased throughout the run. As time proceeds, the pot is gradually depleted of the lighter component. Finally,

a point is reached at which the reflux ratio has attained a very high value. The receivers are then changed, the reflux is reduced, and an intermediate cut is taken as before. This technique can also be extended to a multicomponent mixture.

Variable reflux and constant product composition may be an easier form of control to adopt than the optimal control policy for many batch distillations. It may often be achieved by accurately controlling the column overhead temperature and pressure by use of the reflux flow. This can, however, only be used when the components to be separated are very different in boiling points and when the operating pressure is constant during the run.

In the present study, a PI controller was designed and control parameters were determined for the experimental packed batch distillation column. The controller configuration and design methods applied were given earlier in Chapter 3. Three experiments were carried out using the PI controller, two with the binary system and one with the ternary system. Figure 5.5 shows the performance of the controller when a set point was chosen for 92% Cyclohexane. Figure 5.6 illustrates that the controller was again able to keep the overhead composition constant when the set point was changed to 95%.

The controller was tested for the ternary system and Figure 5.7 shows the profiles of overhead compositions and the reflux ratio.

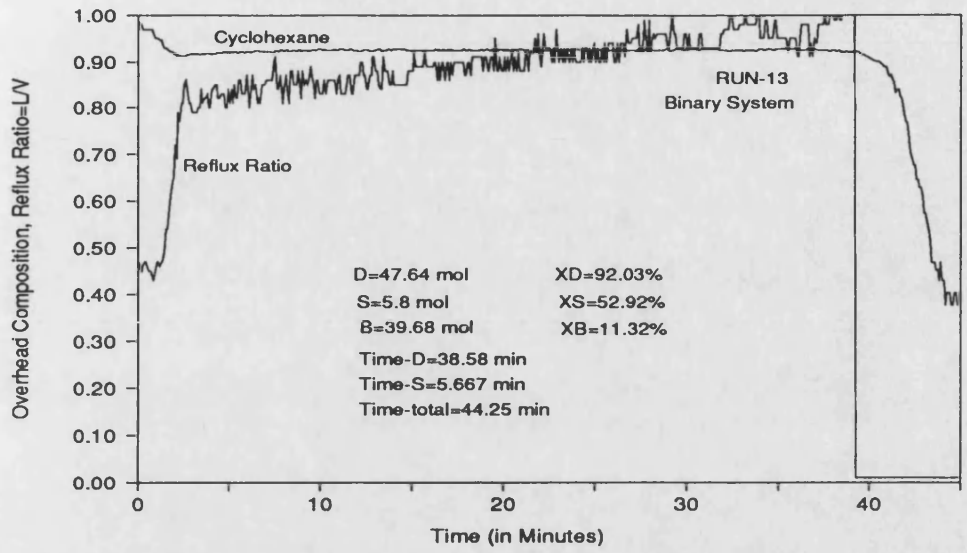


Figure 5.5 Separation profile of controlled overhead composition operation for the first product period of the binary system, Set Point=92%

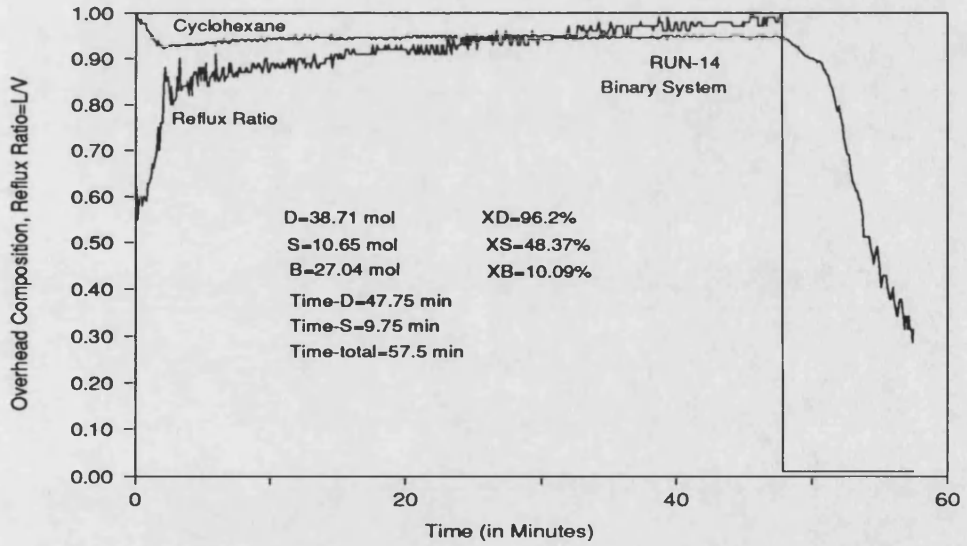


Figure 5.6 Separation profile of controlled overhead composition operation for the first product period of the binary system, Set Point=95%

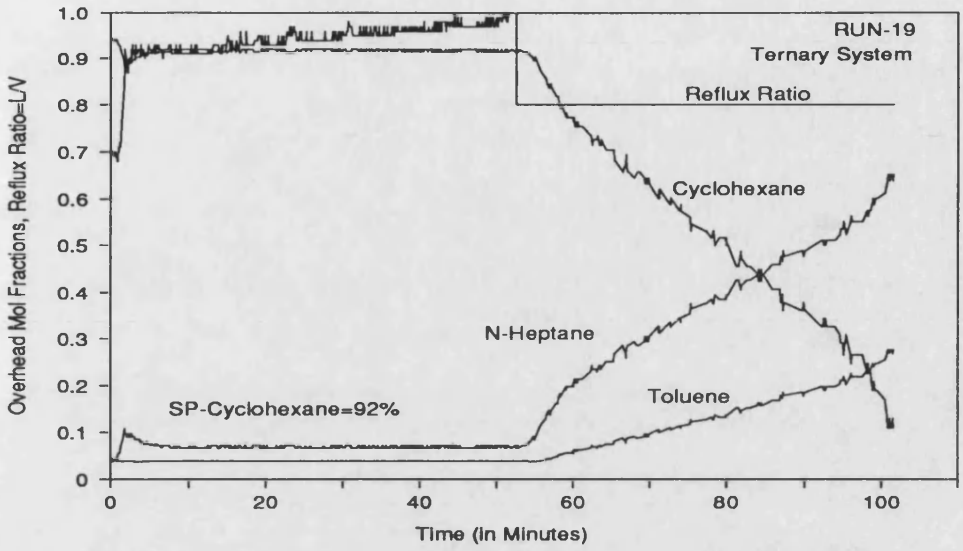


Figure 5.7 Separation profile of controlled overhead composition operation for the first product period of the ternary system, Set Point=92%

It could be concluded from product results that the controlled reflux operation mode gives the highest product amounts with higher purity and the shortest cut point period times with less amount of waste products compared with the constant reflux and piece-wise constant reflux operation modes.

5.3 Cycling operation

A cycling procedure can be used to set the pattern for column operation. The most common form of cycling control is operating the column at total reflux until equilibrium is established. Distillate is then taken as total drawoff for a short period of time, after which the column is again returned to total reflux operation. This cycle is repeated through the course of distillation. An alternative scheme is to interrupt vapour flow to the column periodically by the use of a solenoid operated butterfly valve in the vapour

line from the pot. In both cases, equations necessary to describe the system are very complex, as shown by Schrodt et al., 1967. The most reliable method for establishing the cycle relationships is by experimental trial on an operating column.

5.4 Optimum Operation

The optimal reflux rate policy for the operation of batch distillation columns falls between the two conventional policies and in this neither the reflux ratio nor the instantaneous distillate composition remains constant.

Work on the optimization of multicomponent batch distillation can be divided into two main groups. The first group treats the optimization problem as a control problem or variational problem. While the second group treats it as a design and operation problem or non-linear programming problem.

Early optimization studies have been limited to fairly idealized systems, usually with no tray holdup, no consideration of startup time and a fixed number of trays. Results of these optimization studies have been far from spectacular. Comparisons of batch times with optimal, constant reflux ratio and constant overhead composition control strategies have usually shown only slight differences. Since the fixed reflux ratio policy is the easiest to implement, there appears to be little advantage in using optimal policies.

Control problems of multicomponent batch distillation are mainly concerned with the determination of optimal reflux profile for the operation of an existing column. The choice of an optimization criterion is a rather difficult task, because not only technical but also economic factors have to be taken into account. Furthermore, the problem can be formulated by using several approaches. The first consists of fixing the rectification

time and determining the control variables in order to recover the maximum amount of specified product with a given purity and is also known as the 'Maximum Distillate Problem'. Another approach aims at finding an optimal trajectory to minimize the rectification time for fixed initial and final states and is known as the 'Minimum Time Problem'.

Design and operation problems of multicomponent batch distillation deal with the determination of optimal column size and wider aspects of modes of operation than simple reflux control. Modes of operation pertain to how a column is operated to achieve the desired products. The performance criteria for optimization are usually some formulation of total capacity or profit per cycle time.

The optimization of a batch distillation column is generally considered in the literature as an optimal control policy problem. Especially the determination of optimal reflux policies has induced many researchers to solve one of the following well-known problems:

Table 5.1 Optimization Problems in Literature for the Determination of Optimal Reflux Policies.

Problem	Batch Time t_f	Amount of Distillate $D(t_f)$	Composition of Distillate x_D^*	Objective
Maximum distillate	fixed	free	fixed	maximise D
Minimum time	free	fixed	fixed	minimise t_f
Maximum distillate and minimum time	free	free	fixed	maximise D
Maximum profit	free	free	fixed	maximise profit

5.4.1 The maximum Distillate Problem

It is defined as collection of maximum amount of distillate with specified concentration in a prescribed final time (Converse and Gross,1963; Converse and Huber,1965; Keith and Brunet,1971; Murty et al.,1980; Diwekar et al.,1987; Farhat et al.,1990; Diwekar, 1992)

The work on the maximum distillate problem was first reported by Converse and Gross (1965). They solved the problem using Pontryagin's continuous maximum principle, dynamic programming and calculus of variation. The equations and resulting algorithm developed were limited to the binary, no-holdup case with a constant boilup rate for plate columns. The problem of determining the optimal distillate rate and duration of operation was divided into two parts. First, one can determine the optimal distillate policy for a specified duration of operation. Once this is solved, the overall problem can be

solved by repeating the first step for varies durations. Then by introducing a penalty for longer duration, to account for the increased operating costs, an optimal duration and corresponding distillate rate policy can be determined. They found that the optimal policy resulted in product yields of up to 4 to 5 percent higher than the conventional constant reflux ratio and constant distillate composition policies.

The optimal distillate rate policy, D(t), is defined as that policy which yields the maximum of distillate product of a specified product purity, xD*, in a specified duration, or time, T. The problem can be written as follows.

$$Maximize \quad J = \int_0^T D\,dt \tag{5.1}$$

subject to

$$x_D = \frac{\int_0^T x_D D\,dt}{\int_0^T D\,dt} = x_D^* \tag{5.2}$$

by proper choice of D(t); $0 \le t \le T$.

It is more convenient to modify the product purity constraint, Equation (5.2), to

$$\int_0^T x_D D\,dt = x_D^* \int_0^T D\,dt \tag{5.3}$$

The constraint is then introduced by the method of undetermined multipliers to yield the following problem statement.

$$Maximize \quad P = J - \lambda\left[\int_0^T x_D D\,dt - x_D^* \int_0^T D\,dt\right] \tag{5.4}$$

substituting in Equation (5.1) yields:

$$maximize \quad P = \int_0^T D\{1 - \lambda[x_D^* - x_D]\}dt \tag{5.5}$$

Later work by Converse and Huber (1965) gave an approximate solution for the same problem with plate holdup. It was found that starting from an assumption of all plate compositions being equal to that of the feed, the maximum yield in a fixed time was relatively insensitive to distillate policy, particularly with high feed compositions. However, they concluded that the optimum reflux policy is of the 'bang-bang' type, switching between zero and some specified maximum value.

Optimal policies for a batch packed distillation unit were examined for the cases of variable reflux ratio and variable boilup rate by Keith and Brunet (1971). No-holdup and constant molar overflow assumptions were made. They pointed out that the boilup optimal policy always revealed better performance than the corresponding reflux ratio optimal policy.

Murty et al.(1980) used the same model that was introduced by Converse and Gross (1963) and compared the solution obtained by using different optimization techniques: the dynamic programming, the maximum principle, the quasilinearization technique, the functional conjugate gradient method and the sequential conjugate gradient restoration algorithm. They concluded that dynamic programming conveniently reduces a multistage problem to several single stage problems, but it suffers from the "curse of

dimensionality". Furthermore, for continuous systems, a set of partial differential equations has to be solved. Pontryagin maximum principle applies approximations in function space, but does not suffer from the drawback of an enormous storage requirement as in dynamic programming for a multidimensional problem. In applying the maximum principle to a two-point boundary value problem, however, instability in computation may develop owing to assumed boundary conditions at one end or the other (Pontryagin, et al., 1964).

For solving a nonlinear two-point boundary value problem, the Quasilinearization Technique (QLT) adopts linearization of the system and adjoint equations so that they can be integrated as initial value problems. The quasilinear problem is then solved repeatedly with the hope that it leads to the solution of the original nonlinear problem. This procedure is helpful in overcoming instability; however, linearization has its own price in pot or non-convergence if the initial approximation of a control policy does not lie within an "interval of convergence" (Lee, 1968).

The Functional Conjugate Gradient Method (CGM) proposed by (Lasdon et al., 1967) is a safe and versatile method for handling highly nonlinear chemical engineering problems. It solves a nonlinear problems and develops a quadratic convergence starting from any initial policy far from the optimal. At the same time, it is free from instabilities because of its inherent characteristic of integrating the system and adjoint equations in their natural directions only. In dealing with problems having isoperimetric constraints using any other techniques, one can only resort to a trial-and-error procedure on the Lagrange multipliers which are introduced to account for the constraints. A Sequential

Conjugate Gradient Restoration Algorithm (SCGRA) (Heideman, et al., 1975) fortunately eliminates the necessity for trial-and-error if the problem can be converted into a fixed-end problem. It is also based upon linearization of all the equations.

All of the work reported above is restricted to binary mixtures. This is due to the necessity of solution of an increased number of equations as a result of an increase in the number of components. Another difficulty with their problem formulation is that the number of equations also increases with an increase in the number of plates and hence results in additional memory and computational time requirement. In the work of Diwekar et al. (1987), the problem of an increase in computational time due to the reasons stated above is circumvented by using a short-cut method which is based on an extension of the short-cut method commonly used for continuous columns. The short-cut method used in this study to solve design-optimization procedures is based on three assumptions, namely, zero holdup, constant molar overflow and constant relative volatility. Pontryagin's continuous maximum principle was used to obtain optimal reflux rate policy for the maximum distillate problem. They stated that the method can be applied to columns with large number of plates and a large number of components where the system is near ideal and holdup is small. For highly non-ideal systems, the short-cut method can give a good initial estimate of the trend of reflux policy variation necessary to maintain the desired product compositions.

When multiple-fraction separations are considered, several objectives must be reached so that the conventional approaches based on dynamic programming or variational calculus lead to complex formulations. Indeed, in solving multidimensional problems, the limitations of dynamic programming due to its excessive demand in computer time and storage are well known. For variational calculus, the explicit solution of the

Euler-Lagrange equations can be quite difficult and time-consuming; so from an engineering point of view, it is preferable to convert the variational problem to an approximating nonlinear programming (NLP) problem.

For a multiple-fraction separation problem with fixed final distillation time, the decision variables are the reflux policies for each period and the period switching times. One way to achieve optimization is to repeatedly solve the problem of optimal reflux policies for fixed switching times and then locate an approximation of the optimal solution by iterating the procedure in the time-space. Obviously, such a method would demand excessive computer time.

Logsdon and Biegler (1989) suggested a collocation-based nonlinear programming formulation for obtaining optimal control profiles.

However, when the reflux policies are predefined for each period, the criterion becomes integrable and the problem can be formulated as an NLP problem. The choice of the reflux policy shape can be made in a wide range of functions such as constant, linear, polynomial or exponential and represents the single decision that the operator or the designer must take.

Farhat et al. (1990) is reformulated the problem as an NLP problem by assuming predefined reflux policies. Assumptions of negligible liquid holdup and constant molar flow rates are made to treat multicomponent mixtures.

5.4.1.1 NLP Solution of the Batch Rectification Problem

The batch rectification process described by the DAE's system can be represented by:

$$f[t,x(t),x^{o}(t),R(t),v]=0, \quad t\varepsilon[t_o,t_F] \tag{5.6}$$

where t is the independent variable (time), x(t) ε R^1 is the vector of all state variables (number of moles in the reboiler, compositions and temperatures), $x^{o}(t)$ represents the derivatives of x(t) with respect to time, R(t) ε R is the control variable (reflux ratio), and v ε R^{n+2} is the vector of time invariant parameters (initial compositions, operating pressures, number of sections taken in the column and vapour boilup rate); to simplify the following developments the vector v will not be mentioned. The time interval of interest is $[t_o,t_F]$ where t_o is the initial time and t_F the given final time of the whole operation.

The multiple-fraction batch distillation process is a sequential system described by a series of periods (main cut or off cut), where the initial conditions for each period are the final state of the previous period. For a given period j, the formulation (Eq.(5.6)) becomes:

$$f_j[t,x(t),x^{o}(t),R_j(t)]=0, \quad t\varepsilon[t_{j-1},t_j] \tag{5.7}$$

where t_{j-1} is the initial time of the period j, t_j is the final time of this period, and $R_j(t)$ is the reflux ratio function during this period. For a mixture involving n components, there exists 2n-2 alternate periods productions and off cuts (see Table 5.2), because the heaviest component is recovered into the reboiler; so the index j lies in the interval [1,2n-2]. If

the index j is an odd number i.e., the period $[t_{j-1},t_j]$ is a production period for the key component number k=(j+1)/2, the system is submitted to a purity terminal constraint on the kth key component, this equality constraint occurs when $t=t_j$:

$$g_k[t_j,x(t_j),x^o(t_j),R_j(t)]=0 \tag{5.8}$$

The system performance is measured in terms of the following objective function to be maximized for the main cuts or minimized for the off-cuts:

$$F_j=\int_{t_{j-1}}^{t_j}\frac{V}{R_j(t)+1}dt \tag{5.9}$$

The sequential nature of the batch rectification problem leads to the following multiple criteria decision-making (MCDM) problem:

$$\underset{R_1(t),t_1}{\text{Max}}\,F_1$$

under constraint g_1

$$\underset{R_1(t),R_2(t),t_1,t_2}{\text{Min}}\,F_2$$

under constraint g_1

$$\underset{R_1(t),R_2(t),\ldots,R_j(t),t_1,t_2,\ldots,t_j}{\text{Max}}\,F_j(j\ odd) \tag{5.10}$$

under constraints $g_1,g_2,\ldots,g_{(j+1)/2}$

$$\underset{R_1(t),R_2(t),\ldots,R_{j+1}(t),t_1,t_2,\ldots,t_{j+1}}{\text{Min}}\,F_{j+1}$$

under constraints $g_1,g_2,\ldots,g_{(j+1)/2}$

$$\underset{R_1(t),R_2(t),\ldots,R_{2n-2}(t),t_1,t_2,\ldots,t_{2n-2}=t_F}{\text{Max}} F_{2n-1}$$

under constraints $g_1,g_2,\ldots,g_n$.

At the junction points between the two periods, the purity constraints must be verified, that is why, for example, constraint g_1 is written for period 2 and so on.

The solution of MCDM problem (Eq.(5.10)) is always done by converting it to a scalar optimization problem:

$$\underset{R,T}{\text{Max}} \sum_{j=1}^{2n-1} \beta_j F_j \tag{5.11}$$

under constraints $G=(g_1,g_2,\ldots,g_n)$ where R is the set of the reflux ratio $R_1(t),R_2(t),\ldots,R_{2n-2}(t)$, T is the set of switching times $t_1,t_2,\ldots,t_{2n-2}$, β_j is a positive (respectively negative) factor for a main cut (or off-cut). In practical industrial applications, the weighting factors β_j can be assimilated to economic factors.

In a batch distillation process, the problem generally consists of maximizing the production of main cuts within a given cycle time so that a reduced form of Eq.(5.11) becomes

$$(P1) \quad \underset{R,T}{\text{Max}} \sum_j \beta_j F_j \quad j\epsilon[1,3,5,\ldots,2n-1] \tag{5.12}$$

under the set of constraints G.

When the shape of the reflux policy $R_j(t)$ is chosen in such a way that the objective function F_j (Eq.(5.9)) is integrable in each period j, Eq.(5.12) can be reformulated as an NPL problem. As it is shown in Table 5.2, The functions F_j of the optimization problem (Eq.(5.12)) are the production cuts P_k [k=(j+1)/2].

Another solution of minimizing the weighted sum of off-cuts is

$$(P2) \quad \underset{R,T}{\text{Min}} \sum_j \beta_j F_j \quad j = 2,4,6,\ldots,2n-2 \tag{5.13}$$

under the set of constraints G, where the functions F_j are the off-cuts S_k (k=j/2). In problem (P2), the weighting factors β_j either have an economic significance (recycling or marginal costs) or a technical interpretation (storage requirements).

Table 5.2 Succession of production cuts and off-cuts for a quaternary mixture

Periods	1	2	3	4	5	6
Weighting factors	β_1	β_2	β_3	β_4	β_5	β_6
Cuts	P_1	S_1	P_2	S_2	P_3	S_3 (and P_4 at t_F)
Reflux Policies	R_1	R_2	R_3	R_4	R_5	R_6
Key components	1	-	2	-	3	-

Formulation of problem (P1)

The maximum production-cut formulation is:

$$(P1) \quad \text{Max}\left\{\beta_{2n-1}P_n + \sum_j \beta_j P_k\right\}$$

$$j = 1,3,5,\ldots,2n-3; \quad k = (j+1)/2 \tag{5.14}$$

The last production term has been isolated from the general summation, because it is expressed from the overall mass balance:

$$P_n = B_o - \sum_{i=1}^{n-1}(P_i + S_i)$$

$$= B_o - \sum_j (P_k + S_k)$$

$$= B_o - \sum_j \left[\int_{t_{j-1}}^{t_j} \frac{V}{(R_j(t)+1)}dt + \int_{t_j}^{t_{j+1}} \frac{V}{(R_{j+1}(t)+1)}dt\right]$$

$$j = 1,3,5,\ldots,2n-3; k = \frac{(j+1)}{2} \tag{5.15}$$

Then, the problem (P1) can be written as follows:

$$\text{Max}\left[\beta_{2n-1}B_o + \sum_j \left\{(\beta_j - \beta_{2n-1})\int_{t_{j-1}}^{t_j} \frac{V}{(R_j(t)+1)}dt - \beta_{2n-1}\int_{t_j}^{t_{j+1}} \frac{V}{(R_{j+1}(t)+1)}dt\right\}\right]$$

$$j = 1,3,5,\ldots,2n-3; \quad k = (j+1)/2 \tag{5.16a}$$

with the nonlinear purity constraints (computed from the simulation model):

$$g_k = \overline{X}_{k,D} - \overline{XSPEC}_k$$

that is to say

$$g_k = \frac{\int_{t_{2k-2}}^{t_{2k-1}} \frac{VX_{k,D}(t)}{R_{2k-1}+1} dt}{\int_{t_{2k-2}}^{t_{2k-1}} \frac{V}{R_{2k-1}+1} dt} - \overline{XSPEC}_k = 0 \qquad (5.16b)$$

$$g_n = \overline{X}_{n,1} - \overline{XSPEC}_n = 0 \qquad (5.16c)$$

where $\overline{X}_{n,1} = X_{n,1}(t_F)$.

The linear constraints are related to the switching times

$$t_j \le t_{j+1} \qquad j = 1,2,\ldots,2n-3 \qquad (5.16d)$$

Furthermore, all the variables are physically bounded.

Constant Reflux Policy:

Let $R_j(t) = R_j$ and $R_{j+1}(t) = R_{j+1}$, where R_j and R_{j+1} are constant reflux to be optimized for production and off-cut periods; so the production quantities at the head of the column are integrable:

$$P_k = \int_{t_{j-1}}^{t_j} \frac{V}{(R_j+1)} dt = \frac{V(t_j - t_{j-1})}{(R_j+1)}$$

$$j = 1,3,5,\ldots,2n-3; \quad k = (j+2)/2 \qquad (5.17)$$

and

$$S_k = \int_{t_j}^{t_{j+1}} \frac{V}{(R_{j+1}+1)} dt = \frac{V(t_{j+1}-t_j)}{(R_{j+1}+1)}$$

$$j = 1,3,5,\ldots,2n-3; \quad k = (j+2)/2 \tag{5.18}$$

For all periods j=1,3,5,...,2n-3, problem (P1) becomes:

$$\text{Max}\left[\beta_{2n-1}B_o + \sum_j \left\{(\beta_j - \beta_{2n-1})\frac{V(t_j - t_{j-1})}{(R_j+1)} - \beta_{2n-1}\frac{V(t_{j+1}-t_j)}{(R_{j+1}+1)}\right\}\right]$$

$$j = 1,3,5,\ldots,2n-3; \quad k = (j+1)/2 \tag{5.19}$$

which is a classical NLP form for problem (P1).

Linear Reflux Policy:

An example of formulation of problem (P1), when the reflux policies are assumed to be linear during each production period, is discussed in this section.

$$P_k = \int_{t_{j-1}}^{t_j} V dt/(R_j+1)$$

$$= \int_{t_{j-1}}^{t_j} V dt/(a_j t + b_j + 1) \tag{5.20}$$

where k is the index of the key component which is collected at the end of period j and

$$A = a_1, a_2, \ldots, a_{2n-2}$$

$$B = b_1, b_2, \ldots, b_{2n-2}$$

$$T = t_1, t_2, \ldots, t_{2n-3}$$

When $a_j \neq 0$, after the integration of Eq.(5.16a), problem (P1) becomes

$$(P1) \quad \underset{A,B,T}{\text{Max}}\left\{ \beta_{2n-1}P_n + \sum_j \beta_j P_k \right\}$$

$$j\varepsilon[1,3,5,\ldots,2n-3]; \quad k=(j+1)/2 \tag{5.21}$$

$$\underset{A,B,T}{\text{Max}}\{\beta_{2n-1}B_o + \sum_j [(\beta_j - \beta_{2n-1})V/a_j \ln[(a_j t_j + b_j + 1)/(a_j t_{j-1} + b_j + 1)]$$

$$-\beta_{2n-1}V/a_{j+1}\ln[(a_{j+1}t_{j+1}+b_{j+1}+1)/(a_{j+1}t_j+b_{j+1}+1)]]\}$$

$$j=1,3,5,\ldots,2n-3; \quad k=(j+1)/2 \tag{5.22}$$

Exponential/constant Reflux Policy:

The reflux policy is now assumed to be an exponential function for the production period $R_j(t)=A_j\exp(w_j t)$ and constant for the off-cut period $R_{j+1}(t)=R_{j+1}$. Therefore, the optimization variables are A_j, w_j and R_{j+1}. The integration of the overhead production cut P_k leads to:

$$P_k = \int_{t_{j-1}}^{t_j} \frac{V}{(A_j \exp(w_j t)+1)} dt$$

$$= V\left[(t_j - t_{j-1}) - \frac{1}{w_j} \ln \frac{(1+A_j \exp(w_j t_j))}{(1+A_j \exp(w_j t_{j-1}))} \right]$$

$$j = 1,3,5,\ldots,2n-3; \quad k = (j+2)/2 \tag{5.23}$$

and the integral of S_k is given by Eq.(5.18); so the NLP form for problem (P1) is:

$$\text{Max}[\beta_{2n-1} B_o + \sum_j \{(\beta_j - \beta_{2n-1}) V[(t_j - t_{j-1})$$

$$- \frac{1}{w_j} \ln \frac{(1+A_j \exp(w_j t_j))}{(1+A_j \exp(w_j t_{j-1}))}] - \beta_{2n-1} \frac{V(t_{j+1} - t_j)}{(R_{j+1}+1)} \}]$$

$$j = 1,3,5,\ldots,2n-3; \quad k = (j+1)/2 \tag{5.24}$$

Formulation of problem (P2)

The NLP problem can be solved either by maximizing the weighted sum of production cuts [problem (P1)] or by minimizing the weighted sum of off-cuts [problem (P2)], which are equivalent problems when all the weighting factors are the same. When the reflux policies are assumed to be linear, the problem (P2) can be expressed as follows:

$$(P2) \quad \underset{A,B,T}{\text{Min}}\left\{\sum_j \beta_j S_k\right\}$$

$$j = 2,4,6,\ldots,2n-2; \quad k = (j)/2 \tag{5.25}$$

$$S_k = \int_{t_{j-1}}^{t_j} V dt/(R_j + 1)$$

$$= \int_{t_{j-1}}^{t_j} V dt/(a_j t + b_j + 1) \tag{5.26}$$

When $a_j \neq 0$, problem (P2) becomes:

$$(P2) \quad \underset{A,B,T}{\text{Min}}\left\{\sum_j \beta_j V/a_j \ln[(a_j t_j + b_j + 1)/(a_j t_{j-1} + b_j + 1)]\right\}$$

$$j = 2,4,6,\ldots,2n-2; \quad k = (j)/2 \tag{5.27}$$

under the set of constraints (Eqs.(5.16b,c,d)) (A,B,T are the same vectors as in the previous case).

The problems, (P1) and (P2), have the same dimension, it means the same number of optimization variables, linear constraints, and nonlinear ones. Table 5.3 represents the variation of this dimension as a function of the number of feed components.

Table 5.3 Dimension of Problems (P1) and (P2) for Linear Reflux Policies

No. Of Comp.	2	3	4	5	10
Periods	2	4	6	8	18
Decision Variables	5	11	17	23	53
Linear Constraints	1	2	3	4	9
Nonlinear Constraints	2	3	4	5	10

Luyben (1971,1988) proposed a capacity factor (CAP) which is defined as the total on-specification products produced per unit time. CAP is calculated by dividing the total on-specification products by the total time of the batch, t_F, plus a 30 min period to empty and recharge the still pot. CAP can, in general, be expressed as

$$CAP = \frac{\sum_{j=1}^{NC} P_j}{t_F + 0.5} \tag{5.28}$$

where P_j=total moles of the jth product collected during the batch. The total time of batch includes the startup time at total reflux, the overhead products and slop cuts withdrawal time and the final bottoms purification time. The system is optimized by maximizing CAP. An optimum reflux ratio will correspond to that maximum.

Diwekar (1992) presented a new algorithm for the solution of optimal control problems in a combination of the maximum principle and NLP optimization techniques accompany with a short-cut model of batch distillation. She also showed that different categories of the optimal control problems in batch distillation as maximum distillate, minimum time or optimal profit essentially involve the solution of the maximum distillate problem.

5.4.2 The Minimum Time Problem

It is defined as minimization of final distillation time to produce a given amount of distillate with specified concentration (Price,1967; Coward,1967a,b; Robinson,1969,1970; Mayur and Jackson,1971; Egly et al.,1979,1983; Hansen and Jorgensen,1986; Mujtaba and Macchietto,1988). In other words, the optimal route through a batch distillation process is generally taken to be that which produces the required quantity and quality of product in the shortest possible time. Working from this definition both (Coward, 1967) and (Price, 1967) have presented solutions based on Pontryagin's maximum principle for binary mixtures under conditions of constant boilup rate and when holdup is neglected. Coward compared his optimal route with the equivalent constant reflux and the constant overhead product purity operations. He found that the time saving was small, being only a few percent in each case considered.

Coward (1967) reformulated the problem by changing the independent variable from time to the still composition x_B and presented the solution based on the calculus of variations. He pointed out that several advantages were gained, such as, reduction of the number of differential equations and reduction of the iteration steps.

Robinson (1969) used the maximum principle of Pontryagin to calculate the optimal route under specified conditions: (1) A binary mixture with no holdup and constant boilup rate. (2) A binary mixture with no holdup but the boilup rate is a linear function of the still composition. (3) A multicomponent mixture with no holdup and constant boilup rate. (4) A binary mixture with condenser holdup considered and a constant boilup rate. He pointed out that the requirement of negligible plate holdup is the most difficult assumption to remove because of its effect on the theory. The inclusion of holdup on p

plates would require the introduction of p+3 adjoint parameters to solve a binary problem, and satisfying the boundary conditions of the problem becomes a difficult and time consuming problem.

Robinson (1970) carried out experiments to determine whether such improvements were obtainable in practice on a low holdup industrial still. His experiments on a 30 theoretical stage industrial batch still (47 actual plate) processing a ten component turpentine mixture showed that holdup could not be ignored, even though this totalled less than 4 percent of the initial still charge. The programme was written to solve the multicomponent problem, but the small integration interval necessary to maintain stability through the run was such that excessive computer times were required. Only by treating the problem as a binary mixture could a reflux scheme computed in a realistic time, so the problem was reduced to a pseudo-binary mixture.

Mayur and Jackson (1971) also considered multicomponent separations with negligible holdup by application of the maximum principle as well as the effect of liquid holdup on the optimum policy for binary systems. They stated that a considerable computational effort was involved for only a ternary system and the computation time for this method of solution would be prohibitive for systems of more than three components. Consideration of the plate holdup resulted in a considerable increase in the dimensionality of the problem in proportion to the number of plates. This made the solution based upon the maximum principle computationally impracticable and as a result, the method of steepest-descent was used.

Egly et al. (1979) solved the optimal time problem in multicomponent tray columns with holdup and the possibility of reactions. They transformed the minimal time problem into one with fixed final time and free final conditions which is solved using the modified conjugate gradient method.

Determination of optimal reflux and boilup policies for batch distillation in packed columns has received considerably less attention, although such columns are used very widely, especially for low pressure distillation. Hansen and Jorgensen (1986) applied Pontryagin's maximum principle for binary batch distillation in tray and packed columns. The results for packed columns with constant mass transfer coefficients showed a similar improvement with tray columns when either the reflux ratio or the boilup rate were controlled optimally. For optimal control of both boilup rate and reflux ratio, the time saving was nearly twice the time saving with optimal variation of only one of the controls. The mass transfer rate, however, is often highly dependent on the vapour flow rate.

5.4.3 The Maximum Distillate and Minimum Time Problem

The problem is to collect the maximum amount of distillate with specified concentration in a minimum distillation time (Gangiah and Husain,1974).

Gangiah and Husain (1974) solved the problem for binary systems neglecting holdup using simple gradient method with constant step size and variable step size and functional conjugate gradient method. The functional conjugate gradient method gave a better performance compared with the other two. The procedure used is as follows:

Maximum distillate rate problem is solved for a fixed final time. Then the further step is to consider the time domain as free or unconstrained, in other words, to treat the final time as an independent variable instead of a fixed parameter.

The following iterative algorithm is given for computing optimum final time.

$$\Delta = t_f - t_{f_{\min}}$$

$$t_f^{i+1} = t_f^i + \varepsilon/\Delta^i \tag{5.29}$$

where ε and i are the step size and iteration, respectively, in the time domain. Thus, Δ becomes a direction of search in the time domain and the necessary condition for reaching minimum final time is

$$\Delta^* = maximum\,of\,\Delta \tag{5.30}$$

The computational scheme is, therefore, as follows:

1. Assume an initial duration of time $t_f^i = t_f^o$.
2. Solve the maximum distillate problem for t_f^i.
3. Solve the minimum time problem using maximum distillate rate obtained in step 2.
4. Compute $\Delta^i = t_f^i - t_{f_{\min}}^i$.
5. Repeat steps 2 to 4 with the modified t_f^{i+1} from Eq.(5.29), until $\Delta^i = \Delta^*$ is obtained at which point $t_{f_{\min}} = t_f^{i+1} \approx t_f^i$.

5.4.4 The Maximum Profit Problem

Kerkhof and Vissers (1978) considered a more generalized problem as follows: given the distillate composition and the amount of raw material, maximize a profit function (which is the cost of turnover minus the cost of raw material and the operating cost) by finding the optimum batch time per cycle and optimal reflux policy.

They assumed that every batch distillation cycle is followed by a new one and the average profit per cycle (or per annum) is taken. The formulation given is

$$\overline{P}(t_f) = \frac{C_1 * D(t_f) - C_2 * B_o}{t_f + T_s} - C_3 \tag{5.31}$$

in which $C_1 * D(t_f)$ is the turnover per cycle; $C_2 * B_o$ is the cost of raw material per cycle; C_3 the cost per hour of energy, wages, depreciation and upkeep; and T_s is the start-up and shut-down time. At the optimum profit t_f corresponds to the minimum time for fixed $D(t_f)$ and vice versa. The profit may be maximised by initially guessing the amount of distillate and determining the corresponding minimum distillation time and then continuing to iterate on D until P is maximal.

They defined two dimensionless numbers which play decisive and almost independent roles. The first number, containing only process parameters,

$$q = \frac{X_D^* - X_{Bo}}{X_{Bo}(1 - X_D^*)(\alpha^{N+1} - 1)} \tag{5.32}$$

indicates the degree of difficulty of a separation. When N, α and X_{Bo} become smaller or when the required X_D^* increases the number q grows and the separation becomes more

difficult. The degree of difficulty is an excellent means of systematising all the results. They showed that for easy separations (q smaller than 1%) optimum control and constant overheads were almost the same. When the separation becomes more difficult (q increases up to 20%) optimum control gave up to 5% more distillate or about 40% more profit. Optimum control made more sense when the batch time increased. Optimal control led to much more profit than constant reflux control. Only for very difficult separations (q>15%) the constant reflux control was more favourable than the constant overheads.

The second dimensionless number p,

$$p = \frac{T}{T+T_s}\ \frac{C_2}{C_1}\ \frac{X_D^*}{X_{Bo}} + \frac{T_s}{T+T_s}\ \frac{D(T)}{D_{max}} \tag{5.33}$$

combines the influences of the cost factors and the startup time, and it is a measure of the optimum batch time. Optimum control becomes more attractive when p increases, for instance when the quotient of the prices of raw material and of distillate increases and/or when the startup time becomes longer. The number p can be calculated only if the results of the optimization are known, because T and $D(T)/D_{max}$ are needed. p was approximated by

$$p \approx p_n = \frac{C_2}{C_1} * \frac{x_D^*}{x_{B_o}} + 0.12 \tag{5.34}$$

This expression has the advantage that p_n can be computed without carrying out the optimization. Kerkhof and Vissers (1978) formulated the following rule of thumb from these two dimensionless numbers: The percentage difference in the amount of distillate

obtained between the optimum and constant-overheads policy is $\frac{1}{2} \cdot p_n \cdot q$. The rule is very attractive because both numbers p_n and q can be determined without the need to calculate any control policy.

Kerkhof and Vissers (1978) solved the maximum profit problem by application of the Maximum Principle of Pontryagin for binary batch distillation processes with negligible holdup. It is interesting to consider the explanation of them concerning the difference between the optimum and the constant overhead policies which follows. The constant overhead control requires a low reflux ratio at the beginning of the batch. This causes a high production rate. Going to the end of the batch, the reflux ratio becomes high and the production rate low in consequence; it is then difficult to attain the prescribed average distillate composition. The optimum reflux policy takes this negative effect into account by starting with a somewhat higher reflux ratio so that a higher overhead composition is achieved. This higher value of overhead composition admits a lower overhead composition towards the end of the batch and thus a higher production rate. This foresight on the part of the optimal policy results in a higher average production rate. This effect is more important as the separation can be considered as more difficult.

Diwekar and Madhavan (1986) and Diwekar et al. (1989) considered annual profit as an objective function for development of a short-cut method for design of multicomponent batch distillation processes. Fibonacci search was used to find the optimum solution. The objective function was given as follows:

P=Annual sales value of distillate - annual cost of distillation

Annual cost = Amortized cost of distillation column/year + Amortized cost of reboiler and condenser/year + Annual cost of steam and coolant.

$$= \frac{C_1 VN}{G_a} + \frac{C_2 V}{G_b} + C_3.V.t.NB \qquad (5.35)$$

where C_1 is the amortized incremental investment cost [$/m^2/plate/year]; V is the vapour boilup rate [kg/h]; N is the number of plates; Ga is the allowable vapour velocity [kg/h/m^2]; C_2 is the amortized incremental investment cost in the tubular equipment [$/m^2/year]; Gb is the vapour handling capacity of the tubular equipment [kg/h/m^2]; C_3 is the cost of the steam and coolant to vaporize and condense, respectively, per kg of distillate; t is the time required per batch [h]; and NB is the number of batches per year ($NB = 24x365/(t + t_s)$) ,ts being the time required for startup and shutdown [h].

With the inclusion of cost terms, the objective function becomes

$$P = \sum_{i=1}^{nfr} C(i)D(i)NB - C_1 VN/Ga - C_2 V/Gb - C_3 VtNB \qquad (5.36)$$

where C(i) is the cost of the product for the ith fraction, per kg; and D(i) distillate in fraction i, per batch, [kg/batch].

Logsdon et al. (1990) considered the simultaneous optimal design and operation of batch distillation columns for binary and multicomponent systems. Short-cut and modified short-cut models were verified with a rigourous plate to plate model. The objective function including the capital costs and the utility costs to an hourly basis has the following form:

$$\Psi = \frac{DP_r - B_o C_o}{t_f + T_s} - \frac{T.A.C}{Hrs.} \tag{5.37}$$

where Pr is the sales price of product; Co is the cost of raw material; Hrs. Is the hours per year and T.A.C. Is the total annual cost given by Douglas (1988)

$$T.A.C. = K_1 V^{0.5} N^{0.8} + K_2 V^{0.65} + K_3 V \tag{5.38}$$

The terms in the right hand side of the equation stand for the column, the reboiler and condenser, and the utility costs. The profit function was maximized subject to the differential and algebraic equations used to describe the model. They noted that under certain conditions (small set-up times and capital costs) both equipment parameters N and V have optimal values at their upper bounds.

Jang (1990) defined the problem in a different way to minimize the distillation cost. For a ternary system, the distillation is divided into three periods 0-t1, t1-t2, and t2-t3. The first period collects product one with composition xS1, the third period collects product two with composition xS2. The distillate collected in period two will be recycled. Therefore, the objective function of this example is defined as below:

$$OBJ = COST - BENEFIT \tag{5.39}$$

$$COST = (Fc*Pc + Fstm*Pstm)*t_3 + \left(FEED - \int_{t_1}^{t_2} D\,dt \right)*Praw \tag{5.40}$$

$$BENEFIT = P_{x1}* \int_{0}^{t_1} D\,dt + P_{x2}* \int_{t_2}^{t_3} D\,dt + UN*P_h \tag{5.41}$$

subject to

$$\int_{0}^{t_1} (D*x_1)dt / \int_{0}^{t_1} D\,dt = xS1 \tag{5.42}$$

$$\int_{t_2}^{t_3} (D*x_2)dt / \int_{t_2}^{t_3} D\,dt = xS2 \tag{5.43}$$

The objective of this optimization for the distillation becomes to minimize the objective function. In Eq.(5.39), the variable COST represents the total operation cost of the distillation including the consumption cost of cooling water, steam and raw materials during the distillation. The amount of raw material consumption is computed in the last term in Eq.(5.40) which is the original feed less the recycle distillate. BENEFIT means the benefit from the sales of products of this distillation. The objective function is subject to two constraints. The first constraint xS1 is the predetermined value that the composition of product one must meet. The second constraint xS2 is for the second component.

Jang (1990) solved the optimization problem to find the optimum constant reflux ratio policy through the simulation and piecewise constant reflux ratio policy that each time period will have a different reflux ratio.

The whole simulation program becomes the major part of the objective function subroutine. The main program calls the Hooke-Jeeves subroutine, then the subroutine dominates the optimum search by determining new search variable values for each move. The search variable is reflux ratio or reboiler duty or both. Once the search variable values are determined, the simulation at the specific set of search variables is performed. The simulation is terminated when the distillate composition of second product reaches a predetermined value and the time is assigned as t3. After the simulation, distillate component composition curves and distillate rate curve are saved in matrices. The matrices then are used to perform the fibonacci one dimensional search. The variables t1 and t2 then are determined by the fibonacci optimum search. The purpose of the fibonnacci optimum search is to compute t1 and t2 to meet the two restrictions of the two products specifications. Once t1 and t2 are determined, the objective function value then is computed by utilizing Eqs.(5.39), (5.40) and (5.41). The Hooke-Jeeves subroutine then can check if the move is a success and determine the other set of search variables for next move. The optimization is terminated when the search stepsizes have been reduced below a prespecified value.

Usually, optimum search methods assume that the objective function is a unimodel function and search for the local maximum or minimum only. The method can not guarantee that the search will get the global optimum. Therefore several different sets of initial search variables should be used and the best one should be as the global optimum.

Jang (1990) concluded that the contour of the objective function is very flat for the reboiler duty but the reflux ratio has much more effect on the optimum search. This is because of the fact that, when increasing reboiler duty, the distillation will be accelerated. Thus, the t3 value decreases, but at the same time the steam rate in the reboiler and the

cooling water rate in the overhead condenser increases. The two factors compensate each other and may keep the COST value from changing much as a result of the perturbation of the reboiler duty. The composition curves do not change, the BENEFIT value therefore is not changed. Thus, the objective function value for this search only has a slight change, although the perturbation of the reboiler duty is very big.

For the piecewise constant reflux policy requires more computational effort to perform the Fibonacci searches because the simulation must be performed for each Fibonacci move. Once the Hooke-Jeeve subprogram dominates the optimum search, the simulation then is performed according to the following steps.

1. Determine a new set of search variables R1, R2 and R3 and start the simulation with the reflux ratio R1.
2. Stop and save the distillate composition and rate curves data in arrays when the most volatile component composition reaches a prespecified value which is small enough allowing the Fibonacci method to be used to compute the t1 value.
3. Perform the first Fibonacci search to find t1.
4. After getting t1, restart the simulation from the beginning with the reflux ratio R1 and stop when the simulation time just exceeds t1. Compute by using an interpolation technique and save all the variables required in the simulation at the time t1.
5. Perform the second Fibonacci search to find t2 by the following steps, t2 now is the search variable of the Fibonacci search.
   a) Assume a new t2 value.
   b) Start the simulation at t1 with the reflux ratio R2, and change the reflux ratio from R2 to R3 when the simulation time reaches the assumed t2 value.

c) Stop the simulation when the heavier component composition reaches the predefined value.

d) Compute the objective function to check if the t2 value satisfies the second restriction to meet xS2 specification. If no, go back to step a. If yes, stop the simulation and the t2 value is the correct answer.

The purpose of the Fibonacci search is to compute t1 and t2 to meet the two product specification restrictions as shown in Eqs.(5.42) and (5.43). Equation (5.42) is the restriction that the composition of product one must meet the predetermined specification xS1. To perform the Fibonacci search, Eq.(5.41) must be rearranged and objective function becomes

Objective function : Minimize

$$OBJ1 = \left(\int_0^{t_1} (D x_1) dt / \int_0^{t_1} D\, dt - xS1 \right)^2 \tag{5.44}$$

Search variable : t1

Stopping criterion : OBJ1 < 1.0e-5

After the simulation of each optimum search move, distillate component composition (x1 and x2) and distillate rate (D) are saved in matrices. The search then is performed as follows:

1. A t1 value is determined by the Fibonacci search method.
2. Compute the objective function value by using equation OBJ1.
3. Check the stopping criterion, if satisfied, terminate the search and the t1 is the answer. If not satisfied, repeat the procedure until the stopping criterion is satisfied.

4. If the stopping criterion is not satisfied and the Fibonacci search range is smaller than the prespecified value , terminate the search, the search is not converge.

The search procedure for the other periods is the same as the first one except the different objective function and search variable. For example, for the second product, the objective function is

Objective function : Minimize

$$OBJ2 = \left(\int_{t_2}^{t_3} (Dx_2)dt / \int_{t_2}^{t_3} D\,dt - xS2 \right)^2 \tag{5.45}$$

Search variable : t2

Stopping criterion : OBJ2 < 1.0e-5

5.5 Recycled Waste-Cut

Batch distillation is often used to separate a quantity of material much larger than the capacity of the kettle, by distiling a sequence of charges, and in this case, modes of operation more general. In particular, it is no longer necessary that each charge be separated into a distillate and a residue simultaneously meeting the imposed specifications (see Figure 5.8(a)), since any off-specification material may be returned to the kettle, as a part of the next charge, for reprocessing. This suggests the possibility of periodic operation, in which each charge consists of fresh feedstock mixed with the recycled off-specification material from the previous distillation. Each batch separation, then, consists of two parts:

1) A quantity of accumulated overheads meeting the distillate specification is collected;

2) The residue is further distiled until it meets the bottoms composition specification. The overhead produced during the second part meets neither specification, and is referred to as the 'waste-cut'. It is recycled as part of the charge for the next batch.

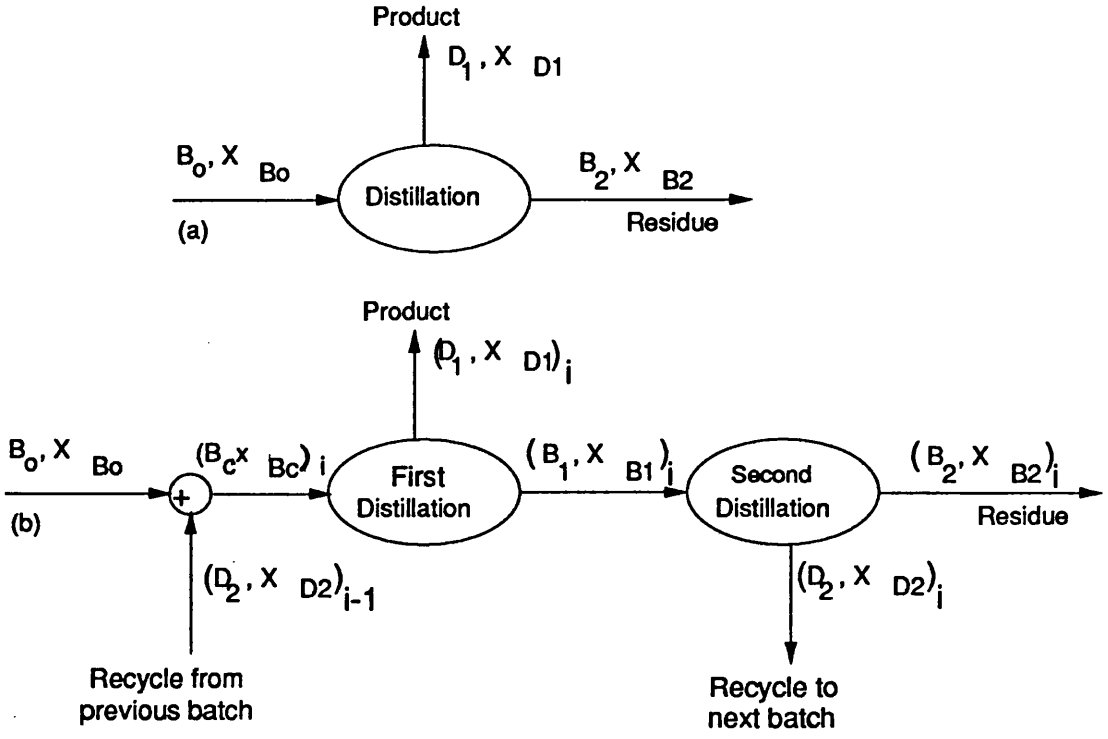


Figure 5.8 Schematic diagrams for binary batch distillation strategies: (a) simple batch distillation; (b) batch distillation with recycled waste cut

If B_o is the amount of fresh feedstock in the charge, and x_{Bo} its composition, the course of the cycling operation is as indicated in Figure 5.8.(b). The feedstock is mixed with a quantity D_2 of the waste-cut, of composition x_{D2}, to form the kettle charge B_c, of composition x_{Bc}. The distillation then proceeds until the composition x_{D1}, of the accumulated distillate D_1 becomes equal to x_D^*, when the distillate product is withdrawn. There remains a quantity B_1 of residue, of composition x_{B1}, in the kettle, and the

distillation is continued until the quantity and composition of the residue become B_2 and x_{B2} respectively, where $x_{B2}=x_B^*$. The accumulated distillate then forms the waste-cut and is mixed with the next batch of fresh feed.

With this kind of operation, it is necessary to consider the choice of reflux ratio during each part of the distillation and a variety of problems results from imposing various constraints on these reflux ratios. Mayur et al. (1970) considered constant reflux, piecewise-constant reflux and piecewise-continuous reflux policies for binary batch distillation columns and applied Pontryagin's maximum principle to find the time optimal reflux policy.

Recycling is particularly interesting if a specified separation is to be performed in an existing column. The number of trays or height of the packing in the column might thus not be appropriate for the new distillation, and running the distillation in a conventional manner without recycling might need a very long distillation time. In such a case recycling offers itself as a possibility of reducing the distillation time. In addition, recycling a waste-cut may be used advantageously to reduce capital investment for a given separation by allowing a smaller column to be used than with ordinary design principles. The economical feasibility will mainly depend on the storage requirements for the waste-cut and upon the energy expenditure necessary to approach the quasi-stationary state compared with the additional capital investment necessary for a column which is sufficiently large to allow the separation be carried out in a single batch distillation.

Christensen and Jorgensen (1987) solved the time optimal problem for a binary batch distillation in a tray column with or without recycling. Results demonstrated that, in

some cases, it was possible to achieve a significant time saving using waste-cut recycling. However, for easy separations, it was not advantageous to use recycling, for difficult separations, the use of recycling meant significant time savings. They used the dimensionless number q (see Eq.(5.32)) given by Kerkhof and Vissers (1978) to demonstrate the time savings obtained using recycled. For $q \leq 0.60$, no time saving was obtained, for $0.60 < q < 0.75$, minor time savings were obtained and $q \geq 0.75$, significant time savings were obtained using recycling.

Luyben (1988) stated that the slop cut in a binary separation can usually be recycled back to the next batch since its composition is often not much different from that of the feed. The slop cut is the distillate that is removed during the period when the overhead contains too much heavy component to be used in the light product and the material left in the still pot and the column still contains too much light component too meet specifications for the heavy product.

For ternary systems, there could be two slop cuts. The first will contain mostly the light component and the intermediate component. The second slop cut will contain mostly the intermediate component and the heavy component. In general, NC-1 slop cuts at most and NC products (NC=number of components) can be obtained.

Marmol and Luyben (1990) treated the operating problem of optimizing the operation of the different slop recycle strategies. They considered only constant reflux ratio. The total batch time was composed of two parts: the operating time and the recharging time. The operating time is the time period while the column is running. The recharging time is the time it takes to empty and recharge the column; this period was assumed constant and equal to 0.5h. The assumptions made in the model were constant tray holdup, constant

reflux drum holdup, equimolal overflow, constant relative volatilities, constant vapour boilup rate and constant reflux ratio. Alternative operating strategies given by Marmol and Luyben (1990) can be divided in two categories, one composed of those that make some kind of mixing of the slop cuts with the next batch (slop recycle, fed-batch and segregated initial charge) and another composed of those that take advantage of the separation already obtained (multicomponent-binary component, intermittent distillate and accumulated product strategy).

1. Total slop recycle: All the slop cuts of the previous batch are combined with fresh feed in the initial charge to the still pot. The effect of total slop recycle of intermediate cuts was studied by Luyben (1988) for ternary case. Although this operation may not be the most efficient, it is certainly the most simple and most widely used in practice. In this scheme, the charge of the first cycle is fresh feed only, and starting from the second cycle, the initial charge is composed of fresh feed and slop cuts of the previous cycle.

2. Multicomponent-Binary Component Batch distillation: A number of slop cuts are saved in segregated tankage and when enough material has accumulated, binary batch distillations on each of the slop cuts are done.

3. Fed-Batch distillation: The slop cuts are fed into the column at an appropriate tray and at an appropriate time during the next batch distillation. The purpose of feeding the slop cuts into the column at some time where the composition profile in some part of the column is similar to that of the slop cut fed is to avoid mixing materials of different compositions. It is assumed that the feeds are at their boiling points. The variables to optimize are reflux ratio, feed plate, feed rate and the time at

which feeding begins.

4. Segregated Initial charge: The second slop cut is fed to the column, while the first is used to fill the reflux drum and, if there is enough material, the column. The purpose of this scheme is to have in the reflux drum and the column, from the beginning, the more volatile components contained in slop cut 1, instead of the multicomponent initial charge. The variables to optimize are reflux ratio, feed plate, feed rate, and the beginning time to feed.

5. Intermittent Distillate: In this case, when the distillate meets specification, the total amount in the reflux drum is withdrawn. It is operated at some fixed reflux and zero distillate flow rate. Then it is operated at total reflux while it is waited for the distillate to meet specification. This scheme is sometimes used in industry because if the analyzed contents of the reflux drum are on-specification, the total amount in it can be discharged to the storage tank, thus decreasing the number of analyzed samples needed during the batch, which is important since usually instantaneous analyzers are not available. The saved slop cuts are treated as they are in multicomponent-binary component scheme. The variables to optimize are the reflux during the building of inventory and the rate of increase of product purity in the reflux drum.

6. Accumulated Product Strategy: Here, the total amount of each product or slop cut in the reflux drum is collected before discharging to the storage tanks. The purpose of this scheme is to reduce the number of samples taken to a minimum to determine if the contents in the reflux drum meet specification. The beginning of product accumulation or the end of slop cut collection is reached when the overhead

vapour composition meets the specified product purity. The variable to optimize is the reflux ratio. Binary distillation is made with each of the slop cuts, the slops of this binary distillation are the first to be saved for the next binary distillation.

Marmol and Luyben (1990) concluded that almost all the strategies (except the segregated scheme and the accumulated strategy for product purity 99%) are better than the total slop recycle that is the most commonly used in practice. The best strategies were multicomponent-binary component and intermittent. The results showed that in general the mixing has a detrimental effect on the rate of product obtained.

5.6 Conclusion

Publications on the operation modes of multicomponent batch distillation were examined and solution methods of different operation problems were revised. It was observed that their applications were mainly on plate type distillation columns using a short-cut method.

All the above modes with the exception of the optimum operation mode can be simulated by the PACBACDIS program. It was found that the controlled mode is the best of the modes among constant reflux, stepwise variable reflux and controlled reflux. As will be mentioned later in Chapter 6, PACBACDIS can easily be enhanced for optimum operation problems with the addition of the required solution routines.

CHAPTER 6

CONCLUSIONS AND RECOMMENDATIONS

Conclusions

Toward this end, some conclusions achieved in this work can be listed as follows:

- A rigourous model was derived for the dynamic behaviour of multicomponent, multipurpose packed batch distillation columns. This model considers reboiler, condenser and packed section liquid and vapour holdups, variable liquid and vapour flow rates, pressure drop and time variant thermodynamic and physical properties. The model takes into account the axial dispersion superimposed on plug flow.

- Experiments were carried out using the pilot scale batch distillation column packed with No.15 Metal Intalox Tower Packing. The column was constructed and automated to study different operation modes of batch distillation columns for this work.

- The development of the simulation program, PACBACDIS, represents a considerable improvement in the modeling of both steady state and dynamic simulation of separation processes in packed batch columns. PACBACDIS was verified extensively using the experimental results. The model predictions compared very well with the experimental data for most of the cases. PACBACDIS gives a very good approximation to the dynamic behaviour of the system.

- The accuracy of predictions for the dynamic column behaviour may be increased by using relatively small time steps in the implicit integration procedure. Using DASSL with variable time steps gives PACBACDIS an additional capacity to handle very stiff problems.

- The relaxation procedure and the Finite Difference Routines that were developed in case of a failure of DASSL increase the stability of the numerical solution of partial and algebraic equation sets. Convergence is assured for physically realizable problems.

- The mathematical model together with the methods of numerical solution developed in this work offers the flexibility to handle a wide variety of industrially significant continuous-contact batch separation processes.

- It could be concluded from the experimental work that the controlled reflux operation mode gives the highest product amounts with higher purity and the shortest cut-point period times with less amount of waste products compared with the constant reflux and piece-wise constant reflux operation modes.

- PACBACDIS can easily be enhanced for optimum operation problems with the addition of the required solution routines. The whole simulation program will be the major part of the objective function subroutine.

Recommendations

Although a number of problems in the area of packed batch distillation column have been solved, there is still considerable room for improvement. A future work is recommended in the following areas:

- The primary area which should be the focus of future work is the computational efficiency of the simulation program to arrive at an accurate solution with a minimum computational time. This is particularly essential for process control purposes. Finite element techniques offer a good alternative to the finite difference method since it requires less number of grid nodes for a given problem area to reach a predefined accuracy.

- More experimental data would be desirable, particularly for multicomponent systems which includes more than three components, using a range of operation modes and operation settings.

- Another area is the solution of optimum control problems for packed batch distillation columns. Work on the optimization of multicomponent batch distillation can be divided into two main groups. The first group treats the optimization problem as a control problem while the second group as a design and operation problem. The control problems of multicomponent batch distillation are mainly concerned with the determination of optimal reflux profile for the operation of an existing column. The design and operation problems deal with the determination of optimal column size and wider aspects of modes of operation than simple reflux control like recycle waste-cut operation. Most of the work reported for optimum operation and design problems is restricted to the batch

distillation in plate columns. A short-cut method which is based on an extension of the short-cut method commonly used for continuous columns is generally used to overcome the problems of memory and computational time requirements of the optimization solution methods applied. A study for an optimum control of packed batch distillation has not been reported in the literature because a more rigourous model should be used and this will increase the complexity of an optimum solution procedure using Pontryagin's Maximum Principle or Nonlinear Programming or the combination of both. A future work on the solution of packed batch optimization problems using direct search methods like Hooke&Jeeves and Fibbonachi Search is required.

REFERENCES

- Aly S. (1985). (Integration of models for packed batch distillation columns by a finite element method). Docteur-Ingenieur Thesis, INP, Toulouse.

- Aly S., Pibouleau L. and Domenech S. (1987). Traitement par une méthode d'éléments finis de modéles de colonne de rectification discontinue á garnissage. Part I: modéle stationnaire avec dispersion axiale. *Entropie*, **136**, 47-57.

- Aly S., Pibouleau L. and Domenech S. (1987). Traitement par une méthode d'éléments finis de modéles de colonne de rectification discontinue á garnissage. Part II: modéle stationnaire avec dispersion axiale et radiale et modéle d'ynamique. *Entropie*, **136**, 58-71.

- Aly S., Pibouleau L. and Domenech S. (1990). Treatment of batch, packed distillation by a finite-element method. Part I. Steady-state model with axial dispersion. *International Chem. Eng.*, **30**(3), 452-463.

- Aly S., Pibouleau L. and Domenech S. (1990). Treatment of batch, packed distillation by a finite-element method. Part II. A steady-state model with axial and radial dispersion and a dynamic model. *International Chem. Eng.*, **30**(3), 464-478.

- API methods, 7A1.1, Page 7-7

- Boston J.F., Britt H.I., Jirapongphan S. and Shah V.B. (1981). An advanced system for the simulation of batch distillation operations. *Foundation of Computer-aided Chemical Process Design*, (Mah R.S.H., Seider W.D. eds.). 203-237.

- Bozenhardt H. (1986). High return strategies for microprocessor based Batch distillation. *Adv. Instrum.*, **41**(1), 77-88.

- Bozenhardt H. (1987). Better sensor/control utilization improves batch distillations. *InTech.*, **34**(3), 43-47.

- Buchanan J.E. (1967). Holdup in irrigated ring-packed towers below the loading point. *Ind. Eng. Chem. Fundam.*, **6**(3), 400-407.

- Christensen F.M. and Jorgensen S.B. (1987). Optimal control of binary batch distillation with recycled waste-cut. *The Chem. Eng. J.*, **34**, 57-64.
- Co P. and Bibaud R. (1971). Longitudinal mixing of the liquid phase in packed columns with countercurrent two phase flow. *The Can. J. of Chem. Eng.*, **49**, 727.
- Coulson J.M. and Richardson J.F. (1978). Chemical Engineering, Unit operations, Volume 2, 3rd edition, Pergamon Press.
- Coulson J.M., Richardson J.F. and Sinnott R.K. (1985). Chemical Engineering, Design, Volume 6. Pergamon Press.
- Converse A.O. and Gross G.D. (1963). Optimal distillate-rate policy in batch distillation. *Ind. Eng. Chem. Fund.*, **2**(3), 217-221.
- Converse A.O. and Huber C.I. (1965). Effect of holdup on batch distillation optimization. *I&EC Fundamentals*, **4**(4), 475-477.
- Coward I. (1967). The time-optimal problem in binary batch distillation. *Chem. Eng. Sci.*, **22**, 503-515 and 1881-1884.
- Cuille P.E. and Reklaitis G.V. (1986). Dynamic simulation of multicomponent batch rectification with chemical reactions. *Comp. Chem. Eng.*, **10**(4), 389-398.
- Danckwerts P.V. (1953). Continuous flow distribution of residence times. *Chem. Eng. Sci.*, **2**, 1-13.
- Deshpande P.B. (1985). Distillation dynamics and control. Research triangle Park, N.C. Instrument Society of America.
- Distefano G.P. (1968). Mathematical modeling and numerical integration of multicomponent batch distillation equation. *A.I.Ch.E J.*, **14**(1), 190-199.
- Diwekar U.M. and Madhavan K.P. (1986). Optimal design of multicomponent batch distillation column. *World Congress III of Chem. Eng.*, Tokyo, 719-722.

- Diwekar U.M., Malik R.K. and Madhavan K.P. (1987). Optimal reflux rate policy determination for multicomponent batch distillation columns. *Comput. Chem. Eng.*, **11**(6), 629-637.
- Diwekar U.M., Madhavan K.P. and Swaney R.E. (1989). Optimization of multicomponent batch distillation columns. *Ind. Eng. Chem. Res.*, **28**, 1011-1017.
- Diwekar U.M. and Madhavan K.P. (1991). Multicomponent batch distillation column design. *Ind. Eng. Chem. Res.*, **30**, 713-721.
- Diwekar U.M. and Madhavan K.P. (1991). BATCH-DIST: A comprehensive package for simulation, design, optimization and optimal control of multicomponent, multifraction batch distillation columns. *Computers Chem. Engng.*, **15**(12), 833-842.
- Diwekar U.M. (1992). Unified approach to solving optimal design-control problems in batch distillation. *AIChE J.*, **38**(10), 1551-1563.
- Domenech S. and Enjalbert M. (1978). Short note, Program for simulation batch rectification as a unit operation. *Computers Chem. Engng.*, **5**(3), 181-184.
- Domenech S. and Enjalbert M. (1981). Program for simulation batch rectification as unit operation. *Comp. Chem. Engng.*, **5**(3), 181-184.
- Dombrowski H.S. and Brownell L.E. (1954). Residual equilibrium saturation of porous media. *Ind. Eng. Chem.*, **46**(6), 1207-1218.
- Douglas J.M. (1988). Conceptual design of chemical processes. McGraw-Hill, New York.
- Dunn W.E. et al. (1977). Longitudinal dispersion in packed gas-absorption columns. *Ind. Eng. Chem. Fundam.*, **16**(1), 116.
- Eckert J.S. (1975). How tower packings behave. *Chem. Eng.*, Albany **82** (April 14), 70.

- Egly H., Ruby V. and Seid B. (1979). Optimum design and operation of batch rectification accompanied by chemical reaction. *Comp. Chem. Eng.*, **3**, 169-174.
- Egly H., Ruby V. and Seid B. (1983). Optimization and control of batch rectification accompanied by chemical reaction. *G. Chem. Eng.*, **6**, 220-227.
- Farhat S., Czernicki M., Pibouleau L. and Domenech S. (1990). Optimization of multiple-fraction batch distillation by nonlinear programming. *AIChE J.*, **36**(9), 1349-1360.
- Fellah M., Domenech S. and Guiglion C. (1982). Modélisation d'une colonne de rectification discontinue á garnissage. Part I: Etablissement des modéles. *Chem. Eng. J.*, **25**, 125-135.
- Fellah M., Domenech S. and Guiglion C. (1982). Modélisation d'une colonne de rectification discontinue á garnissage. Part II: Exploitation numérique. *Chem. Eng. J.*, **25**, 137-146.
- Galindez H. and Frendenslung A. (1988). Simulation of multicomponent batch distillation processes. *Comp. Chem. Engng.*, **12**(4), 281-288.
- Gangiah K. and Husain A. (1974). Variational optimization in control and time domains. *The Canadian J. of Chem. Eng.*, **52**, 654-660.
- Garcia C.E. and Morari M. (1982). Internal model control. 1. A unifying review and some new results. *Ind. Eng. Chem. Process Des. Dev.*, **21**, 308-323.
- Gear G.W. (1971). Simultaneous numerical solution of ordinary differential algebraic equations. *IEEE Trans. On Circuit Theory*, **18**(1), 89-95.
- Gear G.W. (1971). Numerical initial value problems in ordinary differential equations. Prentice-Hall, Englewoog Cliffs, N. J.
- Gear C.W. and Brown R.L. (1973). Documentation for DFASUB-A Program for the solution of Simultaneous Implicit Differantial and Nonlinear Equations. University of Illinois, Dept. of Computer Science,UIUCDCS-R-73-S75.

- Guy J.L. (1983). Modeling batch distillation in multitray columns. *Chem. Eng.*, **10**, 99-103.
- Hansen T.T. and Jorgensen B.S. (1986). Optimal control of binary batch distillation in tray or packed columns. *Chem. Eng. J.*, **33**, 151-156.
- Heideman J.C. and Levy A.V. (1975). *JOTA*, **15**, 203.
- Hindmarsh A.C. (1981). ODE Solvers for use with the Method of Lines. Advances in Computer Methods for Partial Differential Equations-IV, R. Vichnevetsky and R. S. Stepleman, eds., IMACS, New Brunswick, NJ, 312-316.
- Hitch D.M., Rousseau R.W. and Ferrell J. (1986). Simulation of continuous-contact separation processes: multicomponent, adiabatic absorption. *Ind. Eng. Chem. Process Des. Dev.*, **25**, 699-705.
- Hitch D.M. and Rousseau R.W. (1988). Simulation of continuous-contact separation processes: multicomponent batch distillation. *Ind. Eng. Chem. Process Des. Dev.*, **27**, 1466-1473.
- Holland C.D. (1981). Fundamentals of multicomponent distillation. McGraw-Hill.
- Holland C.D. and Liapis A.I. (1983). Computer methods for solving dynamic separation problems. McGraw-Hill.
- Jang M.J. (1990). Dynamic simulation and optimization of multicomponent batch distillation. PhD thesis, Avail. Univ. Microfilms Int., Order No. DA9026755, from: Diss. Abstr. Int. B., 51(5), 2491.
- Keith F.M. and Brunet J. (1971). Optimal operation of a batch packed distillation column. *Can. J. Chem. Eng.*, **49**, 291-295 and 711-712.
- Kerkhof L.H.J. and Vissers H.J.M. (1978). On the profit of optimum control in batch distillation. *Chem. Eng. Sci.*, **33**, 961-970.
- Ki Y.W. (1986). Dynamic matrix control of batch distillation, Ph.D. Thesis, University of Missouri-Columbia.

- King C.J. (1980). Separation processes. 2nd edition, McGraw-Hill, New York, 556-557.
- Kister H.Z. and Gill D.R. (1991). Predict flood point and pressure drop for modern random packings. *Chem. Eng. Prog.*, February, 32-42.
- Kushalkar K.B. and Pangarkar V.G. (1990). Liquid holdup and dispersion in packed columns. *Chem. Eng. Sci.*, **45**(3), 759-763.
- Lambert J.D. (1973). Computational methods in ordinary differential equations. John Wiley & Sons, London.
- Lasdon L.S., Mitter S.K. and Warsen A.D. (1967). *IEEE Trans. Autom. Control*, **AC-12**, 132.
- Lee E.S. (1968). Quasilinearization and invariant imbedding. Academic Press, New York.
- Lee D. (1976). Dynamic and control of feedback multicomponent distillation columns. PhD. Thesis, University of Maryland.
- Leva M. (1953). Tower packings and packed tower design. U.S. Stonware Co.
- Logsdon J.S. and Biegler L.T. (1989). Accurate solution of differential-algebraic optimization problems. *I&EC Research*, **28**, 1628.
- Logsdon J.S., Diwekar U.M. and Biegler L.T. (1990). On the simultaneous optimal design and operation of batch distillation columns. *Trans. IChemE.*, **68**(A), 434-444.
- Luyben W.L. (1971). Some practical aspects of optimal batch distillation design. *Ind. Eng. Chem., Proc. Des. Devel.*, **10**(1), 54-59.
- Luyben W.L. (1988). Multicomponent batch distillation. 1. Ternary systems with slop recycle. *Ind. Eng. Chem. Res.*, **27**, 642-647.
- Luyben W.L. (1990). Process modeling, simulation and control for chemical engineers. 2nd edition, New York, London: McGraw-Hill.

- Marmol E.Q. and Luyben W.L. (1990). Multicomponent batch distillation. 2. Comparison of alternative slop handling and operating strategies. *Ind. Eng. Chem. Res.*, **29**, 1915-1921.

- Mayur D.N., May R.A. and Jackson R. (1970). The time-optimal problem in binary batch distillation with a recycled waste-cut. *Chem. Eng. J.*, **1**, 15-21.

- Mayur D.N. and Jackson R. (1971). Time-optimal problems in batch distillation for multicomponent mixtures and for columns with holdup. *Chem. Eng. J.*, **2**, 150-163.

- Meadows E.L. (1963). multicomponent batch distillation calculation on a digital computer. *Chem. Eng. Progr.*, **59**(46), 48-55.

- Mujtaba I.M. and Macchietto S. (1988). Optimal recycle policies in batch distillation - binary mixtures. *Proceedings SIMO'88*, **2**(6), Toulouse, France, Sept. 14-15, 191.

- Murty B.S.N., Gangiah K. and Husain A. (1980). Performance of various methods in computing optimal control policies. *The Chem. Eng. J.*, **19**, 201-208.

- Nad M. and Spiegel L. (1987). Simulation of batch distillation by computer and comparison with experiment. *CEF 87, XVIII Congress: The Use of Computers in Chemical Engineering EFCE, April 26-30, Glardini Naxos (Italy)*, 737-742.

- Naka U., Komatsu T., Hashimoto I. and Takamatsu T. (1974). Distillate composition change in ternary azeotropic batch distillation. *Kagaku Kogaku*, **38**(7), 501-507.

- Onda K., Takeuchi H. and Okumoto Y. (1968). Mass transfer coefficients between gas and liquid phases in packed columns. *J. of Chem. Eng. of Japan*, **1**(1), 56-62.

- Perry R.H. and Green D. (1984). Perry's chemical engineers handbook. 6th Edition, Mcgraw-Hill.

- Peters W.A. (1922). The efficiency and capacity of fractionating columns. *Ind. Eng. Chem.*, **14**, 476.

- Petzold L.R. (1982). A description of DASSL: A differential/algebraic system solver, SAND82-8637, Sandia National Laboratories, Livermore, CA.

- Price P.C. (1967). The optimization of batch distillation procedures. *Inst. Chem. Engrs. Symposium Series*, **32**(23), 96-104.
- Pontryagin L.S., Boltyanskii V.G., Gamkrelidze R.V. and Mishchenko E.F. (1964). The mathematical theory of optimal processes. (English Translation by Brown D.E.), Macmillan, New York.
- Ramakrishnan R. (1971). Letter to the editor. *Can. J. Chem. Eng.*, **49**, 711.
- Reid R.C., Prausnitz J.M. and Sherwood T.K. (1977). The properties of gases and liquids. 3rd Edition, McGraww-Hill.
- Reuter E., Wozny G. and Jeromin L. (1989). Modeling of multicomponent batch distillation processes with chemical reaction and their control systems. *Comp. Chem. Eng.*, **13**(4/5), 499-510.
- Rippin D.W.T. (1983). Review Paper, simulation of single and multiproduct batch chemical plants for optimal design and operation. *Comput. Chem. Eng.*, **7**(3), 137-156.
- Robinson E.R. (1969). The optimization of batch distillation operations. *Chem. Eng. Sci.*, **24**, 1661-1668.
- Robinson E.R. (1970). The optimal control of an industrial batch distillation column. *Chem. Eng. Sci.*, **25**, 921-928.
- Robinson E.R. (1971). Optimum reflux policies for batch distillation. *Chem. Process. Eng.*, **52**, 47-49.
- Rose L.M. (1985). Distillation design in practice. *Computer-Aided Chemical Engineering*, **1**, Elsevier.
- Sater V.E. and Levenspiel O. (1966). Two-phase flow in packed beds. *I&EC Fundamentals*, **5**(1), 86-92.

- Soderlind G. (1980). DASP3 - A Program for the Numerical Integration of Partitioned Stiff ODE's and Differential/Algebraic Systems. The Royal Institute of Technology, Stockholm, Sweden, Dept. of Numerical Analysis and Computing Science, TRITA-NA-8008.

- Sadotomo H. and Miyahara K. (1983). Calculation procedure for multicomponent batch distillation. *Ind. Chem. Eng.*, **23**(1), 56-64.

- Schrodt V.N., Somerfeld J.T., Martin O.R., Parisot P.E. and Chien H.H. (1967). Plant-scale study of controlled cyclic distillation. *Chem. Eng. Sci.*, **22**, 759-767.

- Sherwood T.K., Pigford R.L. and Wilke C.R. (1975). Mass Transfer.

- Shina N.K. and Kuszta B. (1983). Modeling and identification of dynamic systems. Van Nostrand Reinhold Co., NY.

- Shulman H.L., Ullrich C.F., Proulx A.Z. and Zimmerman J.O. (1955). II. Wetted and effective-interfacial areas, gas-and-liquid phase mass transfer rates. *A.I.Ch.E. Journal*, **1**(2), 253-258.

- Soave G. (1972). Equilibrium constants from a modified Redlich-Kwong equation of state. *Chem. Eng. Sci.*, **27**, 1197-1203.

- Stephanopoulos G. (1984). Chemical process control - An introduction to theory and practice. Prentice-Hall Inc.

- Steward R.R., Weisman E., Goodwin B.M. and Speight C.E. (1973). Effect of design parameters in multicomponent batch distillation. *Ind. Eng. Chem. Process Des. Develop.*, **12**(2), 130-136.

- Stiso M.J. (1969). Packed column dynamics in batch distillation, Ph.D. Thesis, University of Florida.

- Strigle R. (1987). Random packings and packed towers. *Gulf Publ. Co.*, Houston.

- Thoenes D. and Kramers H. (1958). Mass transfer from spheres in various regular packings to a flowing fluid. *Chem. Eng. Sci.*, **8**, 271-283.

- Treybal R.E. (1980). Mass-transfer operations. 3rd edition, McGraw-Hill, New York.

- Wang J. (1986). Some practical aspects of optimal batch distillation operation and control. *World Congress III of Chemical Engineering, Tokyo*, 715-718.

- Weisman E. (1970). Multicomponent batch distillation a theoretical and experimental investigation of constant reflux ratios, step changes in reflux ratios, column holdups, and number of plates. Ph.D. Thesis, Northeastern University.

- Whatley M.J. (1971). A suboptimal control scheme for multicomponent batch distillation. Ph.D. Thesis, University of Florida.

- Wilke C.R. (1950). Diffusional properties of multicomponent gases. *Chem. Eng. Prog.*, **46**, 95.

Appendix 1. Properties of Different Packing

Data listed below for properties of different packings was obtained from Norton Company.

Table A1.1 Packing Factors for Random Dumped Packing

	Nominal Packing Size (inches)							
Packing Type	1/2	5/8	3/4	1	1-1/4	1-1/2	2	3 or 3-1/2
IMPT (Metal)		51		41		24	18	12
Hy-Pak (Metal)				45		29	26	16
Super Intalox Saddles(Ceramic)				60			30	
Super Intalox Saddles(Plastic)				40			28	18
Pall Rings (Plastic)		95		55		40	26	17
Pall Rings (Metal)		81		56		40	27	18
Intalox Saddles (Ceramic)	200		145	92		52	40	22
Raschig Rings (Ceramic)	580	380	255	179	125	93	65	37
Raschig Rings (1/32" Metal)	300	170	155	115				
Raschig Rings (1/6" Metal)	410	300	220	144	110	83	57	32
Berl Saddles (Ceramic)	240		170	110		65	45	

Table A1.2 Norton Tower Packing Surface Areas

PACKING TYPE	MATERIAL	SURFACE AREA M^2/M^3	VOID FRACTION
5/8" Pall Rings	Metal and Plastic	340	0.93
1" Pal Rings	Metal and Plastic	210	0.94
1 1/2" Pall Rings	Metal and Plastic	130	0.96
2" Pall Rings	Metal and Plastic	100	0.96
3" Pall Rings	Metal	65.6	0.95
3 1/2" Pall Rings	Metal	65	
3 1/2" Pall Rings	Plastic	85	
No 1 Super Intalox Saddles	Plastic	210	
No 2 Super Intalox Saddles	Plastic	110	
No 3 Super Intalox Saddles	Plastic	90	
No 1 Super Intalox Saddles	Ceramic	250	
No 2 Super Intalox Saddles	Ceramic	105	
Hypack No 1	Metal	180	0.973
Hypack No 1 1/2	Metal	113	0.966
Hypack No 2	Metal	95	0.98
Hypack No 3	Metal	60	0.982
1/2" Intalox Saddles	Ceramic	620	
3/4" Intalox Saddles	Ceramic	335	
1" Intalox Saddles	Ceramic	255	
1 1/2" Intalox Saddles	Ceramic	195	
2" Intalox Saddles	Ceramic	120	
3" Intalox Saddles	Ceramic	90	
IMTP No 15	Metal	93	0.961
IMTP No 25	Metal	230	0.965
IMTP No 40	Metal	152	0.971
IMTP No 50	Metal	92	0.977
IMTP No 70	Metal	59.4	0.984

Appendix 2. Pressure Drop Correlation for the Packed Columns

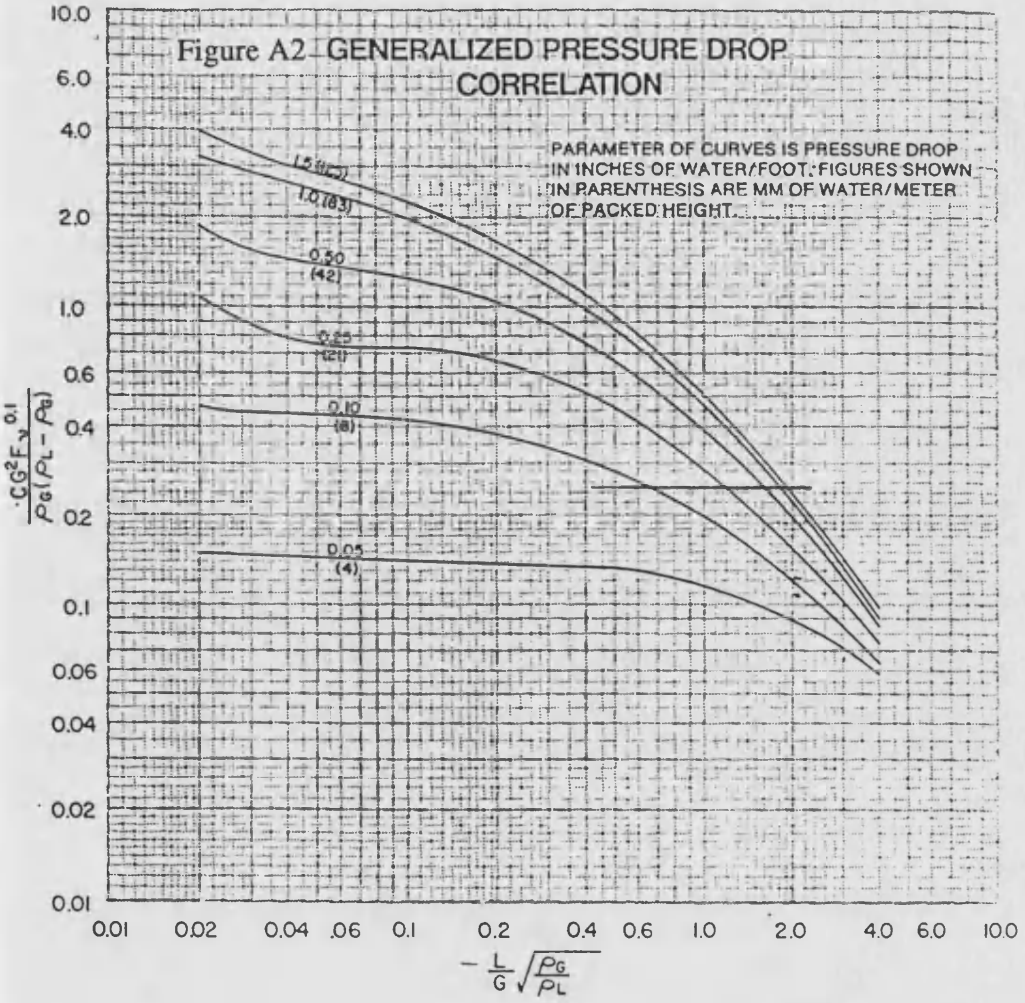
Data used for the pressure drop correlation for the packed columns was obtained from Norton Company. Numerical data was extracted from Figure A2 for the simulation purposes and this data is given in Table A2 Pressure drop is calculated using numerical interpolation and extrapolation from this data.

Table A2 Pressure Drop Correlation Data Derived from Figure A2.

$$Y = \frac{CG^2 Fv^{0.1}}{\rho_G(\rho_L - \rho_G)}$$

$X = \frac{L}{G}\sqrt{\frac{\rho_G}{\rho_L}}$	ΔP=4	ΔP=8	ΔP=21	ΔP=42	ΔP=83	ΔP=125
0.02	0.150	0.470	1.100	1.900	3.25	4.00
0.03	0.148	0.415	0.880	1.560	2.85	3.30
0.04	0.146	0.442	0.780	1.450	2.65	3.00
0.05	0.143	0.440	0.760	1.400	2.42	2.80
0.06	0.142	0.430	0.730	1.370	2.35	2.70
0.10	0.140	0.420	0.720	1.270	1.98	2.20
0.20	0.139	0.370	0.660	1.040	1.50	1.70
0.30	0.136	0.340	0.570	0.900	1.20	1.35
0.40	0.133	0.310	0.520	0.750	0.98	1.10
0.60	0.129	0.260	0.420	0.580	0.75	0.80
1.00	0.115	0.200	0.280	0.380	0.46	0.50
2.00	0.088	0.120	0.151	0.195	0.22	0.23
3.00	0.071	0.850	0.100	0.120	1.35	0.14
4.00	0.058	0.065	0.075	0.085	0.09	0.10

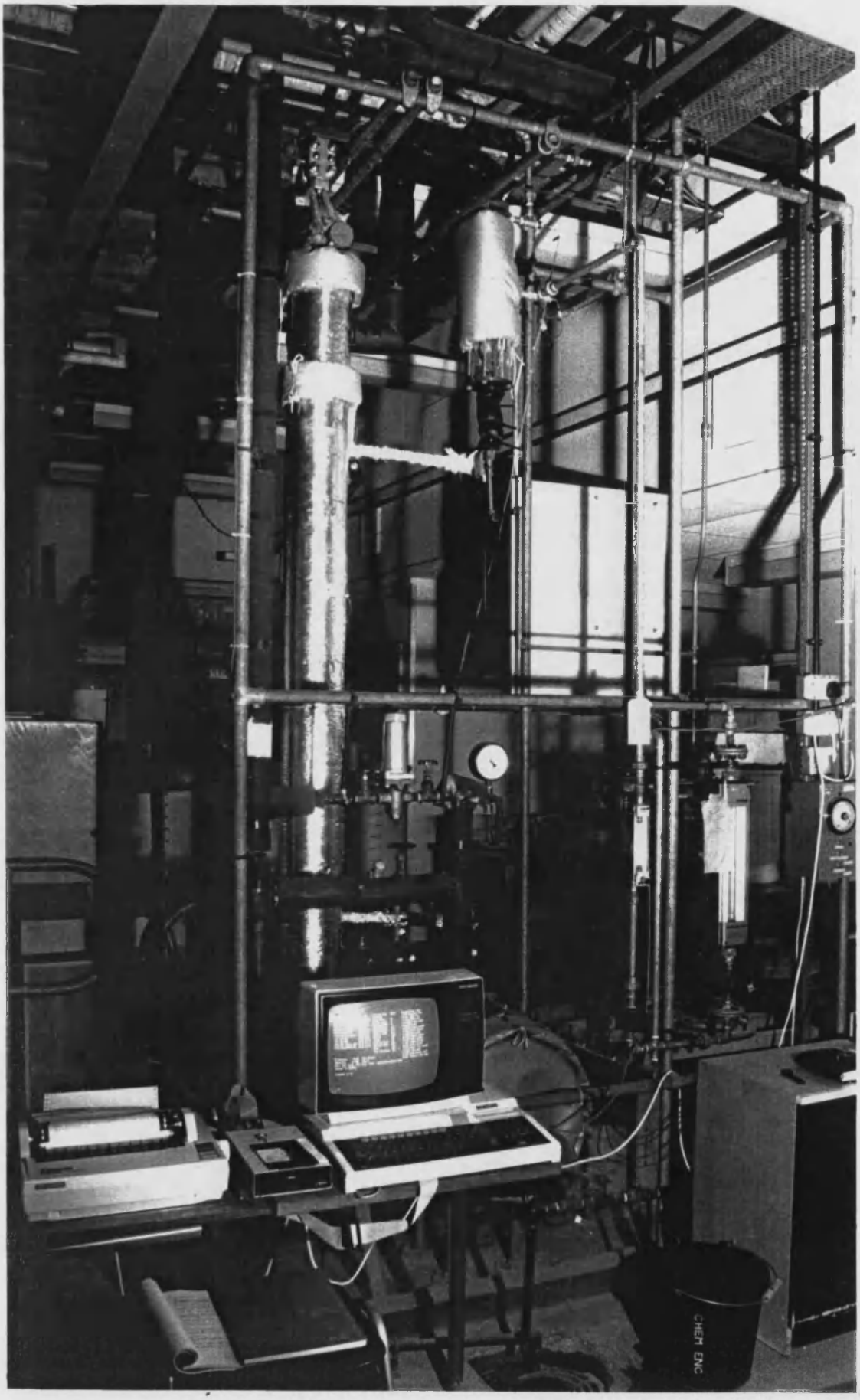
Figure A2 GENERALIZED PRESSURE DROP CORRELATION



GENERALIZED PRESSURE DROP CORRELATION

PROPERTY	SYMBOL	BRITISH UNITS	METRIC UNITS
Gas Rate	G	Lbs/ft² sec	KG/M² s
Liquid Rate	L	Lbs/ft² sec	KG/M² s
Gas Density	ρ_G	Lbs/ft³	KG/M³
Liquid Density	ρ_L	Lbs/ft³	KG/M³
Liquid Viscosity	ν	Centistokes	Centistokes
Conversion Factor	C	1.000	10.764
Packing Factor	F	—	—

Appendix 3. Experimental Packed Batch Distillation System



Appendix 4. BINARY.DAT input file for the binary system runs

```
'INPUT FILE FOR THE BINARY SYSTEM'
'*****************************************************************'
'PARAMETERS OF COLUMN AND PACKING'
'HEIGHT [M] - Height of the Packed Section'
1.7d0
'COLDIA [M] - Diameter of the Column'
8.0d-2
'TYPACK - Type of Packing'
'[1. Pall Rings]'
'[2. Berl Saddles]'
'[3. Raschig Rings]'
1
'SPECAREA [M2/M3] - Specific Surface Area of the Packing'
'Area of Packing / Volume of Packing'
305.0d0
'PACKFACT [in SI units] - Packing Factor of the packing used'
549.0d0
'VOIDFRAC - Void Fraction of Dry Packing [volume voids/volume bed]'
0.961d0
'PACMAT - Packing Material'
'[1-steel / 2-ceramic / 3-plastic / 4-carbon]'
1
'DP [M] - Nominal Size of Packing'
15.0d-3
'PRESBOT [ATM] - Pressure at the Bottom of the Column'
0.9805d0
'PRESTOP [ATM] - Pressure at the Top of the Column'
0.9795d0
'*****************************************************************'
'PARAMETERS OF MIXTURE USED'
'NMC - Number of Components in the Charge'
2
'NAMECOMP - Components in the Mixture from the most volatile to the heaviest'
CYCLOHEXANE
TOLUENE
'CHARGE [KMOL] - Quantity of the Feed'
115.6d-3
'XFEED(NMC) - Charge Composition from the most volatile to the heaviest'
0.4637d0
0.5363d0
'A1 - Coefficient of Ideal Enthalpy Equation (7A1.1-1 in API 7.Volume)'
-6.09373d0
1.43016d0
'B1 - Coefficient of Ideal Enthalpy Equation (7A1.1-1 in API 7.Volume)'
4.06647d0
2.82287d0
'C1 - Coefficient of Ideal Enthalpy Equation (7A1.1-1 in API 7.Volume)'
-5.09267d-2
-3.60099d-02
'D1 - Coefficient of Ideal Enthalpy Equation (7A1.1-1 in API 7.Volume)'
74.30928d0
3.46143d0
'E1 - Coefficient of Ideal Enthalpy Equation (7A1.1-1 in API 7.Volume)'
162.578d0
185.545d0
```

'WME(NMC) [KG/KMOL] - Molecular weight'
84.161d0
92.141d0
'TBP(NMC) [C] - Boiling Point Temperature'
80.75d0
110.65d0
'W(NMC) - Acentric Factor'
0.213d0
0.257d0
'TC(NMC) [K] - Critical Temperature'
553.54d0
591.7d0
'PC(NMC) [ATM] - Critical Pressure'
40.2d0
40.6d0
'VC(NMC) [CM^3/MOL] - Critical Volume'
307.88d0
316.0d0
'SOLPAR(NMC) - Solubility Parameter'
8.196d0
8.915d0
'TDEN(NMC) [C] - Reference Temperature for Liquid Density'
20.0d0
20.0d0
'LDEN(NMC) [KG/M^3] - Liquid Density'
779.0d0
867.0d0
'VISA(NMC) - The Constant of the Liquid Viscosity Equation'
653.62d0
467.33d0
'VISB(NMC) - The Constant of the Liquid Viscosity Equation'
290.84d0
255.24d0
'CPA(NMC) - The Constant of the Ideal Gas Heat Capacity Equation'
-54.541d0
-24.355d0
'CPB(NMC) - The Constant of the Ideal Gas Heat Capacity Equation'
6.112d-1
5.124d-1
'CPC(NMC) - The Constant of the Ideal Gas Heat Capacity Equation'
-2.523d-4
-2.765d-4
'CPD(NMC) - The Constant of the Ideal Gas Heat Capacity Equation'
1.321d-8
4.911d-8
'**********'
'OPERATION PARAMETERS'
'DECIDE - Operation Condition input'
'[1-constant boilup rate]'
'[2-constant heat input rate to Reboiler]'
'[3-variable heat input and boilup rate]'
3
'VBOIL[KMOL/HR] or QREB[KJ/HR] - Boilup Rate of Vapour from Reboiler'
'or Heat Input to Reboiler or initial, rate and maximum values of heat input'
6577.19038d0
10769.0828d0
13102.0d0
'VOLRDRUM [M^3] - Volume of the Reflux Drum'
0.775d-3

'VOLCONHOLD - Volumetric Condenser Holdup [M^3 Liquid/M^3 Condenser]'
0.1d0
'OPTION - Operation Condition'
' [1-Variable Volume Holdup]'
' [2-Constant Volume Holdup]'
' [3-No Holdup]'
2
'If OPTION=2 then enter holdups in [M^3 fluid holdup / M^3 column volume]'
'VOLLHOLD - Constant Liquid Volume Holdup'
0.01d0
'VOLGHOLD - Constant Vapour Volume Holdup'
0.0001d0
'OPMODE - Operation Mode'
' [1-Constant Reflux]'
' [2-Variable Reflux]'
' [3-Constant Overhead Composition]'
' [4-Optimum Operation]'
2
'if OPMOD=1 then'
'REFLUX - Constant Reflux Ratio during first and second products respectively'
2.3333d0
0.0101d0
'if OPMOD=2 then enter VAR - the number of Reflux changes'
8
'Enter Switched time [hr] and Reflux Ratio (=L/D) Respectively'
0.0d0,0.66667d0
0.0625d0,1.0d0
0.1333d0,1.5d0
0.2445d0,2.333d0
0.4375d0,4.0d0
0.5695d0,9.0d0
0.7945d0,19.0d0
0.857d0,0.0101d0
'if OPMOD=3 then'
'XPURITY(NMC) - Desired separation of the components as molar fraction'
0.85d0
0.9180d0
'if OPMOD=4 then'
'OBJFONK - Objective Function Alternatives'
' [1-Maximum Distillate]'
' [2-Minimum Time]'
' [3-Maximum Profit]'
1
'MODELCHOICE - Choice of Model'
'[1] - Plug Flow Model'
'[2] - Backmixing superimposed on Plug Flow Model'
2
'***'
'SIMULATION PARAMETERS'
'DELTAZ [M] - Spatial Step Size=[HEIGHT/(NUMG-1)]'
0.188889d0
'RELAXFAC - Relaxation factor [0<= RELAXFAC <=1]'
0.9D0
'YAZTOT - The Frequency of Output at each iteration level in startup'
150
'YAZPRO - The Frequency of Output at each iteration level in Product Section'
1
'MOLRESULT - File name for Mol Fraction Output'
molfrac.2dyn
molfrac.2tot

'FLOWRESULT - File name for Temperature, Flow rate and Enthalpy Output'
flowtemp.2dyn
flowtemp.2tot
'PHYRESULT - File name for Physical Property Output'
physic.2dyn
physic.2tot
'ERRORRES - File name for Iterative Calculation Report'
norm.2dyn
norm.2tot
'TOTALREF - File name for Total Reflux Condition Steady State Result'
steady.2tot
'PRODUCTRES - Product Period Report'
product.2dyn
'MODELRES - File name for results which are used to compare with experiment'
ap26-2.axial
'SOLSTART - Start the Solution from'
'[1. New Total Reflux Calculations]'
'[2. the Saved File]'
1
'ALFAKY=KY/KYcalculated , parametric study on total gas mass transfer coeff.'
0.0667d0
'ALFAKX=KX/KXcalculated , parametric study on total gas mass transfer coeff.'
17.50d0
'BETA1=DVaxial/DVaxialcalculated, parametric study on gas disper. coeff.'
0.0001d0
'BETA2=DLaxial/DLaxialcalculated, parametric study on liquid disper. coeff.'
0.0001d0
'SIMPRO - Choose the method for product section simulation'
'[1-Finite Difference with fixed parameters]'
'[2-DASSL with variable parameters]'
2
'SIMSTOP - Option of the run [1-Only Total Reflux / 2-Product Period too]'
2
'ERRORNORM - Tolerance error value for calculations'
1.0d-4
'STOPCR - Stopping criterio of the calculations'
'[1- Time]'
'[2- Composition]'
1
'ELAPSED - [minutes] Elapsed time for each product period'
51.42d0
61.25d0

TERNARY.DAT input file for the ternary system runs

```
'INPUT FILE FOR THE TERNARY SYSTEM'
'*****************************************************************'
'PARAMETERS OF COLUMN AND PACKING'
'HEIGHT [M] - Height of the Packed Section'
1.70d0
'COLDIA [M] - Diameter of the Column'
8.0d-2
'TYPACK - Type of Packing'
'[1. Pall Rings]'
'[2. Berl Saddles]'
'[3. Raschig Rings]'
1
'SPECAREA [M2/M3] - Specific Surface Area of the Packing'
'Area of Packing / Volume of Packing'
305.084d0
'PACKFACT [in SI units] - Packing Factor of the Packing used'
549.0d0
'VOIDFRAC - Void Fraction of Dry Packing [volume voids/volume bed]'
0.961d0
'PACMAT - Packing Material'
'[1-steel / 2-ceramic / 3-plastic / 4-carbon]'
1
'DP [M] - Nominal Size of Packing'
15.0d-3
'PRESBOT [ATM] - Pressure at the Bottom of the Column'
0.98d0
'PRESTOP [ATM] - Pressure at the Top of the Column'
0.979d0
'*****************************************************************'
'PARAMETERS OF MIXTURE USED'
'NMC - Number of Components in the Charge'
3
'NAMECOMP - Components in the Mixture from the most volatile to the heaviest'
CYCLOHEXANE
N-HEPTANE
TOLUENE
'CHARGE [KMOL] - Quantity of the Feed'
98.26d-3
'XFEED(NMC) - Charge Composition from the most volatile to the heaviest'
0.36742d0
0.28428d0
0.3483d0
'A1 - Coefficient of Ideal Enthalpy Equation (7A1.1-1 in API 7.Volume)'
-6.09373d0
7.50209d0
1.43016d0
'B1 - Coefficient of Ideal Enthalpy Equation (7A1.1-1 in API 7.Volume)'
4.06647d0
3.34199d0
2.82287d0
'C1 - Coefficient of Ideal Enthalpy Equation (7A1.1-1 in API 7.Volume)'
-5.09267d-2
-3.99022d-02
-3.60099d-02
```

'D1 - Coefficient of Ideal Enthalpy Equation (7A1.1-1 in API 7.Volume)'
74.30928d0
14.12428d0
3.46143d0
'E1 - Coefficient of Ideal Enthalpy Equation (7A1.1-1 in API 7.Volume)'
162.578d0
141.338d0
185.545d0
'WME(NMC) [KG/KMOL] - Molecular weight'
84.161d0
100.21d0
92.141d0
'TBP(NMC) [C] - Boiling Point Temperature'
80.75d0
98.45d0
110.65d0
'W(NMC) - Acentric Factor'
0.213d0
0.351d0
0.257d0
'TC(NMC) [K] - Critical Temperature'
553.54d0
540.14d0
591.7d0
'PC(NMC) [ATM] - Critical Pressure'
40.2d0
27.05d0
40.6d0
'VC(NMC) [CM^3/MOL] - Critical Volume'
307.88d0
427.29d0
316.0d0
'SOLPAR(NMC) - Solubility Parameter'
8.196d0
7.43d0
8.915d0
'TDEN(NMC) [C] - Reference Temperature for Liquid Density'
20.0d0
20.0d0
20.0d0
'LDEN(NMC) [KG/M^3] - Liquid Density'
779.0d0
684.0d0
867.0d0
'VISA(NMC) - The Constant of the Liquid Viscosity Equation'
653.62d0
436.73d0
467.33d0
'VISB(NMC) - The Constant of the Liquid Viscosity Equation'
290.84d0
232.53d0
255.24d0
'CPA(NMC) - The Constant of the Ideal Gas Heat Capasity Equation'
-54.541d0
-5.146d0
-24.355d0
'CPB(NMC) - The Constant of the Ideal Gas Heat Capasity Equation'
6.112d-1
6.761d-1
5.124d-1

'CPC(NMC) - The Constant of the Ideal Gas Heat Capasity Equation'
-2.523d-4
-3.65d-4
-2.765d-4
'CPD(NMC) - The Constant of the Ideal Gas Heat Capasity Equation'
1.321d-8
7.657d-8
4.911d-8
'***'
'OPERATION PARAMETERS'
'DECIDE - Operation Condition input'
'[1-constant boilup rate]'
'[2-constant heat input rate to Reboiler]'
'[3-variable heat input and boilup rate]'
3
'VBOIL[KMOL/HR] or QREB[KW/HR] - Boilup Rate of Vapour from Reboiler'
'or Heat Input to Reboiler or initial, rate and maximum values of heat input'
5560.92d0
8997.84d0
13585.0d0
'VOLRDRUM [M^3] - Volume of the Reflux Drum'
0.775d-3
'VOLCONHOLD - Volumetric Condenser Holdup [M^3 Liquid/M^3 Condenser]'
0.1d0
'OPTION - Operation Condition'
' [1-Variable Volume Holdup]'
' [2-Constant Volume Holdup]'
' [3-No Holdup]'
2
'If OPTION=2 then enter holdups in [M^3 fluid holdup / M^3 column volume]'
'VOLLHOLD - Constant Liquid Volume Holdup'
0.01d0
'VOLGHOLD - Constant Vapour Volume Holdup'
0.0001d0
'OPMODE - Operation Mode'
' [1-Constant Reflux]'
' [2-Variable Reflux]'
' [3-Constant Overhead Composition]'
' [4-Optimum Operation]'
1
'if OPMODE=1 then'
'REFLUX - Constant Reflux Ratio of each product period respectively'
2.3333d0
2.3333d0
2.3333d0
'if OPMODE=2 then enter VAR - the number of reflux changes'
1
'Enter Switched Time [hr] and Reflux Ratio (=L/D) respectively'
300.0d0,2.333d0
'if OPMODE=3 then'
'XPURITY(NMC) - Desired Separation of the components as molar fraction'
0.12908d0
0.32256d0
0.78838d0
'if OPMODE=4 then'
'OBJFONK - Objective Function Alternatives'
' [1-Maximum Distillate]'
' [2-Minimum Time]'
' [3-Maximum Profit]'
1

```
'MODELCHOICE - Choice for Model'
'[1] - Plug Flow Model'
'[2] - Backmixing Superimposed on Plug Flow Model'
2
'*************************************************************'
'SIMULATION PARAMETERS'
'DELTAZ [M] - Spatial Step Size=[HEIGHT/(NUMG-1)]'
0.188889d0
'RELAXFAC - Relaxation factor [0<= RELAXFAC <=1]'
0.9D0
'YAZTOT - The Frequency of Output at each iteration level in Startup'
50
'YAZPRO - The Frequency of Output at each iteration level in Product Section'
3
'MOLRESULT - File name for Mol Fraction Output'
molfrac.8ter
molfrac.8tot
'FLOWRESULT - File name for Temperature, Flow rate and Enthalpy Output'
flowtemp.8ter
flowtemp.8tot
'PHYRESULT - File name for Physical Property Output'
physic.8ter
physic.8tot
'ERRORRES - File name for Iterative Calculation Report'
norm.8ter
norm.8tot
'TOTALREF - File name for Total Reflux Condition Steady State Result'
steady.8tot
'PRODUCTRES - Product Period Report'
product.8ter
'MODELRES - File name for results which are used to compare with experiment'
june4.8ax
'SOLSTART - Start the Solution from'
'[1. New Total Reflux Calculations]'
'[2. the Saved File]'
1
'ALFAKY=KY/KYcalculated- parametric study on gas mass transfer coefficient'
0.0667d0
'ALFAKX=KX/KXcalculated- parametric study on liquid mass transfer coefficient'
17.5d0
'BETA1=DVaxial/DVaxialcalculated- parametric study on gas phase disper.coef.'
0.0001d0
'BETA2=DLaxial/DLaxialcalculated- parametric study on gas phase disper.coef.'
0.0001d0
'SIMPRO - Choose the method for product section simulation'
'[1-Finite Difference with fixed parameters]'
'[2-DASSL with variable parameters]'
2
'SIMSTOP - Option for the run [1-Steady state calculations only/2-Product too]'
2
'ERRORNORM - Tolerance error value for calculations'
1.0d-4
'STOPCR - Stopping criterio of the calculations'
'[1- Time]'
'[2- Concentration]'
1
'ELAPSED - [minutes] Elapsed time for each product period'
65.5d0
65.5d0
65.5d0
```

Appendix 5. Extracts from report files that were produced by PACBACDIS

File=MOLRESULT

time [h]= 0.0221175031
iteration no= 3

component=CYCLOHEXANE

node	Z [m]	x	xint	yint	y
1	0.0000000	0.454310140	0.454310140	0.517578182	0.517578178
2	0.1214290	0.459012159	0.458592583	0.617490933	0.527199344
3	0.2428580	0.473118166	0.472677841	0.630727554	0.537165454
4	0.3642870	0.487482519	0.487018946	0.643959862	0.547416149
5	0.4857160	0.502249376	0.501760630	0.657309990	0.557932646
6	0.6071450	0.517390882	0.516874770	0.670739108	0.568694518
7	0.7285740	0.532876575	0.532330609	0.684208339	0.579679812
8	0.8500030	0.548674035	0.548095394	0.697679527	0.590865311
9	0.9714320	0.564749226	0.564134725	0.711115682	0.602226795
10	1.0928610	0.581066737	0.580412777	0.724481262	0.613739298
11	1.2142900	0.597590089	0.596892607	0.737742472	0.625377357
12	1.3357190	0.614282215	0.613536618	0.750867644	0.637115296
13	1.4571480	0.631105889	0.630306968	0.763827491	0.648927508
14	1.5785770	0.648024061	0.647165935	0.776595286	0.660788977
15	1.7000060	0.664999266	0.664999130	0.664742800	0.664742800

component=N-HEPTANE

node	Z [m]	x	xint	yint	y
1	0.0000000	0.270345738	0.270345738	0.248854308	0.248854308
2	0.1214290	0.269030069	0.269177622	0.216406368	0.246161953
3	0.2428580	0.265083078	0.265240517	0.211013012	0.243245394
4	0.3642870	0.260880850	0.261048925	0.205492534	0.240120736
5	0.4857160	0.256381453	0.256560745	0.199798169	0.236792169
6	0.6071450	0.251590966	0.251782131	0.193948638	0.233265102
7	0.7285740	0.246517479	0.246721255	0.187963615	0.229546130
8	0.8500030	0.241170854	0.241388066	0.181863284	0.225642948
9	0.9714320	0.235562570	0.235794141	0.175668039	0.221564258
10	1.0928610	0.229705599	0.229952550	0.169398262	0.217319658
11	1.2142900	0.223614210	0.223877672	0.163074049	0.212919521
12	1.3357190	0.217303722	0.217584939	0.156714930	0.208374871
13	1.4571480	0.210790329	0.211090666	0.150339674	0.203697254
14	1.5785770	0.204090955	0.204411882	0.143966131	0.198898487
15	1.7000060	0.197223608	0.197223477	0.197298898	0.197298898

component=TOLUENE

node	Z [m]	x	xint	yint	y
1	0.0000000	0.275344122	0.275344122	0.233567510	0.233567513
2	0.1214290	0.271957772	0.272229795	0.166102699	0.226638703
3	0.2428580	0.261798757	0.262081641	0.158259434	0.219589152
4	0.3642870	0.251636631	0.251932128	0.150547604	0.212463115
5	0.4857160	0.241369171	0.241678624	0.142891842	0.205275185
6	0.6071450	0.231018152	0.231343099	0.135312254	0.198040380
7	0.7285740	0.220605946	0.220948136	0.127828046	0.190774057
8	0.8500030	0.210155111	0.210516539	0.120457190	0.183491741
9	0.9714320	0.199688204	0.200071135	0.113216278	0.176208947
10	1.0928610	0.189227664	0.189634673	0.106120476	0.168941044
11	1.2142900	0.178795701	0.179229721	0.099183478	0.161703122
12	1.3357190	0.168414064	0.168878443	0.092417425	0.154509833
13	1.4571480	0.158103782	0.158602367	0.085832835	0.147375238
14	1.5785770	0.147884984	0.148422183	0.079438582	0.140312536
15	1.7000060	0.137777126	0.137777394	0.137958302	0.137958302

time [h]= 0.4423500612
iteration no= 60

component=CYCLOHEXANE

node	Z [m]	x	xint	yint	y
1	0.0000000	0.305354998	0.305354998	0.368730614	0.368730753
2	0.1214290	0.309430920	0.309157282	0.458757542	0.377081319
3	0.2428580	0.321658686	0.321374134	0.472904346	0.385961378
4	0.3642870	0.334468913	0.334172509	0.487462968	0.395342907
5	0.4857160	0.347994908	0.347686343	0.502549486	0.405224389
6	0.6071450	0.362233574	0.361912405	0.518122122	0.415600451
7	0.7285740	0.377175933	0.376841556	0.534132888	0.426461680
8	0.8500030	0.392807141	0.392458752	0.550528471	0.437794572
9	0.9714320	0.409106415	0.408742986	0.567251078	0.449581577
10	1.0928610	0.426047153	0.425667393	0.584239436	0.461801238
11	1.2142900	0.443597158	0.443199477	0.601429883	0.474428426
12	1.3357190	0.461718987	0.461301461	0.618757496	0.487434643
13	1.4571480	0.480370430	0.479930756	0.636157256	0.500788410
14	1.5785770	0.499505094	0.499040563	0.653565173	0.514455910
15	1.7000060	0.519072420	0.519072317	0.519011744	0.519011744

component=N-HEPTANE

node	Z [m]	x	xint	yint	y
1	0.0000000	0.319241565	0.319241565	0.303396328	0.303396285
2	0.1214290	0.318607840	0.318689891	0.287088850	0.302097787
3	0.2428580	0.316706666	0.316794207	0.282469934	0.300536197
4	0.3642870	0.314454182	0.314547348	0.277509486	0.298708666
5	0.4857160	0.311819495	0.311918278	0.272168569	0.296607506
6	0.6071450	0.308791957	0.308896364	0.266459232	0.294226732
7	0.7285740	0.305363529	0.305473594	0.260397602	0.291562240
8	0.8500030	0.301528944	0.301644736	0.254003586	0.288611934
9	0.9714320	0.297285827	0.297407460	0.247300528	0.285375816
10	1.0928610	0.292634826	0.292762470	0.240314809	0.281856036
11	1.2142900	0.287579725	0.287713611	0.233075406	0.278056894
12	1.3357190	0.282127440	0.282267872	0.225613345	0.273984798
13	1.4571480	0.276287910	0.276435272	0.217961080	0.269648170
14	1.5785770	0.270073954	0.270228705	0.210151875	0.265057254
15	1.7000060	0.263501188	0.263501326	0.263526949	0.263526949

component=TOLUENE

node	Z [m]	x	xint	yint	y
1	0.0000000	0.375403437	0.375403437	0.327873058	0.327872961
2	0.1214290	0.371961240	0.372152827	0.254153608	0.320820894
3	0.2428580	0.361634648	0.361831659	0.244625720	0.313502425
4	0.3642870	0.351076905	0.351280144	0.235027546	0.305948428
5	0.4857160	0.340185597	0.340395380	0.225281945	0.298168105
6	0.6071450	0.328974469	0.329191230	0.215418646	0.290172817
7	0.7285740	0.317460537	0.317684850	0.205469510	0.281976080
8	0.8500030	0.305663916	0.305896512	0.195467943	0.273593494
9	0.9714320	0.293607758	0.293849553	0.185448394	0.265042607
10	1.0928610	0.281318021	0.281570138	0.175445755	0.256342726
11	1.2142900	0.268823117	0.269086912	0.165494712	0.247514680
12	1.3357190	0.256153573	0.256430667	0.155629160	0.238580559
13	1.4571480	0.243341659	0.243633972	0.145881665	0.229563420
14	1.5785770	0.230420952	0.230730732	0.136282952	0.220486835
15	1.7000060	0.217426392	0.217426357	0.217461307	0.217461307

time [h]= 0.8625826194
iteration no= 117

component=CYCLOHEXANE

node	Z [m]	x	xint	yint	y
1	0.0000000	0.020954452	0.020954452	0.031305149	0.026169823
2	0.1214290	0.021332089	0.021190879	0.039797536	0.032090713
3	0.2428580	0.022465001	0.022310241	0.041861667	0.032994548
4	0.3642870	0.023763753	0.023595025	0.044221311	0.034032780
5	0.4857160	0.025255187	0.025072260	0.046922872	0.035224577
6	0.6071450	0.026966702	0.026769502	0.050012808	0.036591631
7	0.7285740	0.028929294	0.028717915	0.053542850	0.038158420
8	0.8500030	0.031177915	0.030952619	0.057570267	0.039952454
9	0.9714320	0.033751830	0.033513038	0.062158037	0.042004527
10	1.0928610	0.036694961	0.036443239	0.067374900	0.044348940
11	1.2142900	0.040056200	0.039792238	0.073295230	0.047023689
12	1.3357190	0.043889669	0.043614244	0.079998682	0.050070603
13	1.4571480	0.048254899	0.047968840	0.087569569	0.053535403
14	1.5785770	0.053216902	0.052921055	0.096095908	0.057467718
15	1.7000060	0.058778708	0.058845870	0.058778490	0.058778490

component=N-HEPTANE

node	Z [m]	x	xint	yint	y
1	0.0000000	0.293119981	0.293119981	0.311570899	0.304908388
2	0.1214290	0.294482791	0.294324604	0.348408081	0.314405827
3	0.2428580	0.298571222	0.298412487	0.352379328	0.317328835
4	0.3642870	0.302771395	0.302612280	0.356392207	0.320335661
5	0.4857160	0.307090740	0.306931490	0.360442192	0.323422916
6	0.6071450	0.311524287	0.311365103	0.364510725	0.326586062
7	0.7285740	0.316065413	0.315906445	0.368575896	0.329819235
8	0.8500030	0.320705593	0.320546931	0.372612063	0.333115056
9	0.9714320	0.325434134	0.325275794	0.376589525	0.336464428
10	1.0928610	0.330237882	0.330079803	0.380474190	0.339856321
11	1.2142900	0.335100920	0.334942954	0.384227293	0.343277552
12	1.3357190	0.340004251	0.339846155	0.387805200	0.346712558
13	1.4571480	0.344925491	0.344766920	0.391159303	0.350143194
14	1.5785770	0.349838568	0.349679078	0.394236081	0.353548586
15	1.7000060	0.354683812	0.354713265	0.354683717	0.354683717

component=TOLUENE

node	Z [m]	x	xint	yint	y
1	0.0000000	0.685925567	0.685925567	0.657123952	0.668921789
2	0.1214290	0.684185120	0.684484517	0.611794383	0.653503460
3	0.2428580	0.678963777	0.679277272	0.605759006	0.649676617
4	0.3642870	0.673464852	0.673792695	0.599386482	0.645631559
5	0.4857160	0.667654073	0.667996250	0.592634937	0.641352507
6	0.6071450	0.661509011	0.661865395	0.585476467	0.636822307
7	0.7285740	0.655005294	0.655375640	0.577881254	0.632022345
8	0.8500030	0.648116492	0.648500451	0.569817670	0.626932490
9	0.9714320	0.640814036	0.641211168	0.561252438	0.621531045
10	1.0928610	0.633067157	0.633476957	0.552150910	0.615794739
11	1.2142900	0.624842880	0.625264809	0.542477477	0.609698760
12	1.3357190	0.616106080	0.616539602	0.532196118	0.603216839
13	1.4571480	0.606819611	0.607264240	0.521271128	0.596321402
14	1.5785770	0.596944530	0.597399867	0.509668011	0.588983696
15	1.7000060	0.586537480	0.586440866	0.586537793	0.586537793

File= FLOWRESULT

time [h]= 0.0221175031
iteration no= 3

node	Z [m]	temp [C]	L [kmol/m^2 h]	L [kg/m^2 h]	V [kmol/m^2 h]	V [kg/m^2 h]
1	0.0000000	94.73135	27.23770	2470.37831	39.70137	3573.86708
2	0.1214290	91.97058	27.41101	2484.77764	40.22127	3616.70614
3	0.2428580	91.55433	27.93095	2527.87553	40.28209	3618.02313
4	0.3642870	91.14036	27.99186	2529.23045	40.34368	3619.23756
5	0.4857160	90.72532	28.05354	2530.47943	40.40676	3620.42069
6	0.6071450	90.31012	28.11671	2531.69387	40.47123	3621.56920
7	0.7285740	89.89571	28.18127	2532.87015	40.53694	3622.67978
8	0.8500030	89.48304	28.24707	2534.00463	40.60378	3623.74924
9	0.9714320	89.07307	28.31398	2535.09379	40.67158	3624.77472
10	1.0928610	88.66676	28.38186	2536.13443	40.74022	3625.75359
11	1.2142900	88.26502	28.45056	2537.12372	40.80953	3626.68341
12	1.3357190	87.86878	28.51994	2538.05900	40.87937	3627.56191
13	1.4571480	87.47891	28.58983	2538.93784	40.94959	3628.38717
14	1.5785770	87.09625	28.66008	2539.75821	41.01640	3628.83627
15	1.7000060	86.70119	28.72692	2540.19780	41.03867	3628.98201

time [h]= 0.4423500612
iteration no= 60

node	Z [m]	temp [C]	L [kmol/m² h]	L [kg/m² h]	V [kmol/m² h]	V [kg/m² h]
1	0.0000000	99.33048	44.89624	4143.03523	65.36176	5990.1853
2	0.1214290	96.79564	45.14403	4164.20197	66.10492	6053.1955
3	0.2428580	96.26072	45.88739	4227.59027	66.19748	6056.1461
4	0.3642870	95.72593	45.98020	4230.60458	66.29373	6059.0108
5	0.4857160	95.18905	46.07668	4233.52846	66.39471	6061.8792
6	0.6071450	94.65108	46.17787	4236.45091	66.50040	6064.7453
7	0.7285740	94.11320	46.28375	4239.36516	66.61074	6067.6027
8	0.8500030	93.57670	46.39425	4242.26373	66.72562	6070.4442
9	0.9714320	93.04300	46.50926	4245.13854	66.84489	6073.2620
10	1.0928610	92.51362	46.62863	4247.98086	66.96836	6076.0479
11	1.2142900	91.99017	46.75218	4250.78153	67.09581	6078.7933
12	1.3357190	91.47429	46.87966	4253.53109	67.22696	6081.4892
13	1.4571480	90.96770	47.01083	4256.21980	67.36152	6084.1264
14	1.5785770	90.47210	47.14537	4258.83777	67.49562	6086.3766
15	1.7000060	89.97731	47.27941	4261.05652	67.54032	6087.1179

time [h]= 0.8625826194
iteration no= 117

node	Z [m]	temp [C]	L [kmol/m² h]	L [kg/m² h]	V [kmol/m² h]	V [kg/m² h]
1	0.0000000	108.90761	62.32565	5879.73769	89.52513	8451.64197
2	0.1214290	107.95709	62.40775	5887.98087	89.77142	8476.38397
3	0.2428580	107.51309	62.65404	5912.71825	89.81091	8481.58320
4	0.3642870	107.08513	62.69353	5917.91998	89.85259	8486.95494
5	0.4857160	106.67354	62.73521	5923.29418	89.89664	8492.50057
6	0.6071450	106.27845	62.77927	5928.84218	89.94330	8498.22265
7	0.7285740	105.89987	62.82592	5934.56655	89.99281	8504.12354
8	0.8500030	105.53763	62.87544	5940.46959	90.04547	8510.20537
9	0.9714320	105.19141	62.92810	5946.55341	90.10161	8516.47001
10	1.0928610	104.86068	62.98423	5952.81983	90.16157	8522.91906
11	1.2142900	104.54473	63.04420	5959.27037	90.22578	8529.55386
12	1.3357190	104.24265	63.10841	5965.90630	90.29469	8536.37546
13	1.4571480	103.95331	63.17732	5972.72855	90.36880	8543.38464
14	1.5785770	103.67540	63.25143	5979.73775	90.44365	8550.10777
15	1.7000060	103.39486	63.32627	5986.47897	90.46860	8552.34868

File= PHYRESULT

time [h]= 0.0221175031
iteration no= 3

component=CYCLOHEXANE

node	Z [m]	k	liqden [kg/m^3]	gasden [kg/m^3]	liqvis [kg/m hr]	gasvis [kg /m hr]
1	0.0000000	1.4415129	638.30822	2.8479611	1.2180474	0.0301608
2	0.1214290	1.3464913	660.29166	2.8661631	1.2563145	0.0299272
3	0.2428580	1.3343709	653.30114	2.8648906	1.2622388	0.0298920
4	0.3642870	1.3222481	652.84421	2.8639628	1.2681719	0.0298570
5	0.4857160	1.3100071	652.38789	2.8634209	1.2741620	0.0298219
6	0.6071450	1.2976820	651.92167	2.8632605	1.2801964	0.0297867
7	0.7285740	1.2853071	651.44686	2.8634769	1.2862618	0.0297517
8	0.8500030	1.2729162	650.96474	2.8640653	1.2923441	0.0297168
9	0.9714320	1.2605423	650.47651	2.8650206	1.2984288	0.0296821
10	1.0928610	1.2482173	649.98328	2.8663381	1.3045014	0.0296477
11	1.2142900	1.2359719	649.48607	2.8680128	1.3105470	0.0296137
12	1.3357190	1.2238351	648.98581	2.8700399	1.3165505	0.0295802
13	1.4571480	1.2118341	648.48332	2.8724147	1.3224973	0.0295472
14	1.5785770	1.1999941	647.97930	2.8751327	1.3283729	0.0295149
15	1.7000060	1.1877012	647.49511	2.8783675	1.3344796	0.0294815

node	Z [m]	liqdiff [m^2/hr]	gasdiff [m^2/hr]	kx [kmol/m^2 hr]	ky [kmol/m^2 hr]
1	0.0000000	0.0000927	0.0147325	34.2540655	0.1419557
2	0.1214290	0.0000902	0.0145283	34.1445259	0.1425794
3	0.2428580	0.0000895	0.0145102	34.9536553	0.1424325
4	0.3642870	0.0000888	0.0144904	35.6557934	0.1422785
5	0.4857160	0.0000880	0.0144685	36.3623026	0.1421187
6	0.6071450	0.0000872	0.0144446	37.0711133	0.1419534
7	0.7285740	0.0000864	0.0144187	37.7799977	0.1417827
8	0.8500030	0.0000857	0.0143909	38.4867902	0.1416069
9	0.9714320	0.0000849	0.0143614	39.1894148	0.1414260
10	1.0928610	0.0000840	0.0143300	39.8859048	0.1412405
11	1.2142900	0.0000832	0.0142970	40.5744218	0.1410504
12	1.3357190	0.0000824	0.0142623	41.2532767	0.1408560
13	1.4571480	0.0000816	0.0142261	41.9209452	0.1406575
14	1.5785770	0.0000808	0.0141883	42.5760769	0.1404465
15	1.7000060	0.0000800	0.0141544	43.2036668	0.1403305

component=N-HEPTANE

node	Z [m]	k	liqden [kg/m^3]	gasden [kg/m^3]	liqvis [kg/m hr]	gasvis [kg /m hr]
1	0.0000000	0.8685652	604.44345	3.4433014	0.7333061	0.0259082
2	0.1214290	0.8039538	594.36729	3.4666325	0.7486204	0.0257075
3	0.2428580	0.7955535	593.16085	3.4652137	0.7509773	0.0256772
4	0.3642870	0.7871802	591.95884	3.4642183	0.7533341	0.0256471
5	0.4857160	0.7787558	590.74847	3.4636983	0.7557098	0.0256170
6	0.6071450	0.7703034	589.53280	3.4636481	0.7580994	0.0255868
7	0.7285740	0.7618461	588.31481	3.4640617	0.7604974	0.0255567
8	0.8500030	0.7534063	587.09735	3.4649329	0.7628983	0.0255267
9	0.9714320	0.7450060	585.88314	3.4662555	0.7652965	0.0254969
10	1.0928610	0.7366662	584.67472	3.4680232	0.7676862	0.0254674
11	1.2142900	0.7284069	583.47442	3.4702297	0.7700615	0.0254382
12	1.3357190	0.7202471	582.28439	3.4728689	0.7724168	0.0254094
13	1.4571480	0.7122043	581.10654	3.4759348	0.7747463	0.0253811
14	1.5785770	0.7042943	579.94253	3.4794219	0.7770444	0.0253533
15	1.7000060	0.6961083	578.73097	3.4835510	0.7794295	0.0253246

node	Z [m]	liqdiff [m^2/hr]	gasdiff [m^2/hr]	kx [kmol/m^2 hr]	ky [kmol/m^2 hr]
1	0.0000000	0.0000883	0.0137002	21.4528369	0.1206803
2	0.1214290	0.0000859	0.0135253	20.9208109	0.1212851
3	0.2428580	0.0000852	0.0135239	20.5079429	0.1212508
4	0.3642870	0.0000845	0.0135210	19.9823621	0.1212115
5	0.4857160	0.0000838	0.0135164	19.4387254	0.1211684
6	0.6071450	0.0000831	0.0135100	18.8789238	0.1211213
7	0.7285740	0.0000823	0.0135020	18.3047969	0.1210704
8	0.8500030	0.0000816	0.0134923	17.7182554	0.1210157
9	0.9714320	0.0000808	0.0134809	17.1212531	0.1209573
10	1.0928610	0.0000801	0.0134680	16.5157641	0.1208953
11	1.2142900	0.0000793	0.0134535	15.9037560	0.1208298
12	1.3357190	0.0000785	0.0134376	15.2871595	0.1207610
13	1.4571480	0.0000777	0.0134202	14.6678475	0.1206891
14	1.5785770	0.0000770	0.0134014	14.0476202	0.1206067
15	1.7000060	0.0000762	0.0133751	13.4238079	0.1205391

component=TOLUENE

node	Z [m]	k	liqden [kg/m^3]	gasden [kg/m^3]	liqvis [kg/m hr]	gasvis [kg /m hr]
1	0.0000000	0.6589871	744.44185	3.1392899	0.9901184	0.0292836
2	0.1214290	0.6101562	739.51761	3.1597958	1.0122608	0.0290569
3	0.2428580	0.6038555	739.02201	3.1584276	1.0156715	0.0290227
4	0.3642870	0.5975721	738.52367	3.1574419	1.0190826	0.0289887
5	0.4857160	0.5912473	738.01606	3.1568851	1.0225219	0.0289547
6	0.6071450	0.5848986	737.50060	3.1567521	1.0259821	0.0289206
7	0.7285740	0.5785432	736.97869	3.1570377	1.0294552	0.0288866
8	0.8500030	0.5721982	736.45160	3.1577365	1.0329334	0.0288527
9	0.9714320	0.5658801	735.92056	3.1588428	1.0364083	0.0288191
10	1.0928610	0.5596048	735.38666	3.1603513	1.0398717	0.0287858
11	1.2142900	0.5533875	734.85087	3.1622562	1.0433150	0.0287528
12	1.3357190	0.5472423	734.31407	3.1645524	1.0467300	0.0287203
13	1.4571480	0.5411826	733.77701	3.1672344	1.0501083	0.0286883
14	1.5785770	0.5352204	733.24028	3.1702972	1.0534419	0.0286569
15	1.7000060	0.5290478	732.67489	3.1739361	1.0569022	0.0286245

node	Z [m]	liqdiff [m^2/hr]	gasdiff [m^2/hr]	kx [kmol/m^2 hr]	ky [kmol/m^2 hr]
1	0.0000000	0.0001980	0.0147946	32.2205406	0.1364577
2	0.1214290	0.0001927	0.0146141	31.2321995	0.1372052
3	0.2428580	0.0001911	0.0146215	29.9166998	0.1372235
4	0.3642870	0.0001895	0.0146278	28.4751884	0.1372387
5	0.4857160	0.0001879	0.0146325	27.0416703	0.1372519
6	0.6071450	0.0001863	0.0146357	25.6201316	0.1372630
7	0.7285740	0.0001846	0.0146374	24.2141878	0.1372718
8	0.8500030	0.0001829	0.0146376	22.8272588	0.1372782
9	0.9714320	0.0001812	0.0146363	21.4625374	0.1372823
10	1.0928610	0.0001795	0.0146335	20.1229762	0.1372839
11	1.2142900	0.0001778	0.0146293	18.8112736	0.1372830
12	1.3357190	0.0001761	0.0146236	17.5298502	0.1372796
13	1.4571480	0.0001743	0.0146166	16.2808290	0.1372738
14	1.5785770	0.0001726	0.0146081	15.0660283	0.1372570
15	1.7000060	0.0001709	0.0145834	13.8825628	0.1372066

mixture properties

node	Z [m]	Temp [C]	pres [atm]	Liq.ax.Dis. [m^2/hr]	Vap.ax.Dis. [m^2/hr]
1	0.0000000	94.73135	0.99176568	0.0000587	0.09585
2	0.1214290	91.97058	0.99002975	0.0000594	0.09720
3	0.2428580	91.55433	0.98841527	0.0000605	0.09764
4	0.3642870	91.14036	0.98692350	0.0000610	0.09771
5	0.4857160	90.72532	0.98555818	0.0000614	0.09776
6	0.6071450	90.31012	0.98431999	0.0000620	0.09779
7	0.7285740	89.89571	0.98320957	0.0000625	0.09782
8	0.8500030	89.48305	0.98222757	0.0000630	0.09782
9	0.9714320	89.07308	0.98137460	0.0000636	0.09781
10	1.0928610	88.66676	0.98065129	0.0000641	0.09778
11	1.2142900	88.26503	0.98005824	0.0000647	0.09774
12	1.3357190	87.86879	0.97959604	0.0000652	0.09769
13	1.4571480	87.47892	0.97926524	0.0000658	0.09762
14	1.5785770	87.09626	0.97906639	0.0000664	0.09752
15	1.7000060	86.70119	0.97900000	0.0000670	0.09732

node	Z [m]	Liq.Enthalpy [kJ/kmol]	Vap.Enthalpy [kJ/kmol]
1	0.0000000	33296.3270	64288.8883
2	0.1214290	32713.2781	63767.2267
3	0.2428580	32538.0217	63595.8296
4	0.3642870	32359.6626	63420.5373
5	0.4857160	32176.5960	63241.0508
6	0.6071450	31989.1919	63057.7399
7	0.7285740	31797.8552	62871.0015
8	0.8500030	31603.0173	62681.2533
9	0.9714320	31405.1304	62488.9291
10	1.0928610	31204.6637	62294.4748
11	1.2142900	31002.0986	62098.3433
12	1.3357190	30797.9216	61900.9901
13	1.4571480	30592.6190	61702.8676
14	1.5785770	30386.6728	61504.4178
15	1.7000060	30176.6361	61399.0891

node	Z [m]	liqden [kg/m^3]	gasden [kg/m^3]	liqvis [kg /m hr]	gasvis [kg /m hr]
1	0.0000000	658.376295374	3.064159064	1.003023352	0.030502360
2	0.1214290	664.102138847	3.080524328	1.030621676	0.030316214
3	0.2428580	659.800583925	3.075373964	1.039084016	0.030304093
4	0.3642870	658.520475303	3.070450087	1.047783951	0.030293206
5	0.4857160	657.252684095	3.065802823	1.056808212	0.030283279
6	0.6071450	655.995478459	3.061433186	1.066148276	0.030274346
7	0.7285740	654.752534310	3.057342615	1.075793332	0.030266438
8	0.8500030	653.527289334	3.053533043	1.085730526	0.030259580
9	0.9714320	652.322899318	3.050006891	1.095945052	0.030253796
10	1.0928610	651.142217828	3.046767008	1.106420184	0.030249103
11	1.2142900	649.987787324	3.043816636	1.117137379	0.030245513
12	1.3357190	648.861824554	3.041159388	1.128076490	0.030243036
13	1.4571480	647.766200669	3.038799204	1.139215971	0.030241674
14	1.5785770	646.702431071	3.036740194	1.150533037	0.030241429
15	1.7000060	645.669022240	3.038545675	1.162250217	0.030221379

node	Z [m]	liqsurten [n/m]	intarea [m^2/m^3]	liqholdup [kmol]	gasholdup [kmol]
1	0.0000000	0.015351713	249.052884437	0.000042579	0.000000002
2	0.1214290	0.015624421	247.334742358	0.000042972	0.000000002
3	0.2428580	0.015651460	247.821060799	0.000042762	0.000000002
4	0.3642870	0.015679426	247.617712814	0.000042749	0.000000002
5	0.4857160	0.015708514	247.403794684	0.000042740	0.000000002
6	0.6071450	0.015738673	247.180851196	0.000042734	0.000000002
7	0.7285740	0.015769841	246.949284052	0.000042731	0.000000002
8	0.8500030	0.015801950	246.709556761	0.000042731	0.000000002
9	0.9714320	0.015834924	246.462189833	0.000042735	0.000000002
10	1.0928610	0.015868681	246.207758321	0.000042743	0.000000002
11	1.2142900	0.015903136	245.946882656	0.000042753	0.000000002
12	1.3357190	0.015938200	245.680215760	0.000042768	0.000000002
13	1.4571480	0.015973782	245.408437150	0.000042785	0.000000002
14	1.5785770	0.016009789	245.132249242	0.000042806	0.000000002
15	1.7000060	0.016048171	244.833580401	0.000042830	0.000000002

time= 0.4423500612
iteration no= 60

component=CYCLOHEXANE

node	Z [m]	k	liqden [kg/m^3]	gasden [kg/m^3]	liqvis [kg/m hr]	gasvis [kg /m hr]
1	0.0000000	1.5712015	641.62652	2.8941447	1.1580480	0.0305501
2	0.1214290	1.4838967	665.16254	2.8992293	1.1905568	0.0303355
3	0.2428580	1.4715070	658.63150	2.8886463	1.1975910	0.0302903
4	0.3642870	1.4587164	658.31630	2.8790572	1.2046855	0.0302450
5	0.4857160	1.4454103	657.97356	2.8705672	1.2118711	0.0301996
6	0.6071450	1.4316229	657.59147	2.8631968	1.2191353	0.0301540
7	0.7285740	1.4173937	657.17062	2.8569653	1.2264634	0.0301085
8	0.8500030	1.4027676	656.71176	2.8518906	1.2338381	0.0300631
9	0.9714320	1.3877940	656.21571	2.8479897	1.2412400	0.0300180
10	1.0928610	1.3725256	655.68334	2.8452784	1.2486474	0.0299732
11	1.2142900	1.3570184	655.11555	2.8437713	1.2560367	0.0299289
12	1.3357190	1.3413300	654.51324	2.8434817	1.2633827	0.0298852
13	1.4571480	1.3255188	653.87726	2.8444217	1.2706587	0.0298424
14	1.5785770	1.3096434	653.20838	2.8466023	1.2778371	0.0298004
15	1.7000060	1.2933646	652.51940	2.8501347	1.2850641	0.0297586

node	Z [m]	liqdiff [m^2/hr]	gasdiff [m^2/hr]	kx [kmol/m^2 hr]	ky [kmol/m^2 hr]
1	0.0000000	0.0002479	0.0147395	52.8231786	0.2069131
2	0.1214290	0.0002423	0.0146111	54.1751617	0.2077233
3	0.2428580	0.0002405	0.0146349	55.9772516	0.2075469
4	0.3642870	0.0002386	0.0146539	57.6094713	0.2073592
5	0.4857160	0.0002366	0.0146673	59.3023865	0.2071615
6	0.6071450	0.0002346	0.0146750	61.0512780	0.2069538
7	0.7285740	0.0002325	0.0146769	62.8504877	0.2067361
8	0.8500030	0.0002303	0.0146730	64.6937623	0.2065084
9	0.9714320	0.0002281	0.0146631	66.5743546	0.2062710
10	1.0928610	0.0002259	0.0146472	68.4851467	0.2060240
11	1.2142900	0.0002236	0.0146253	70.4187832	0.2057677
12	1.3357190	0.0002213	0.0145975	72.3678108	0.2055023
13	1.4571480	0.0002189	0.0145637	74.3248185	0.2052284
14	1.5785770	0.0002166	0.0145240	76.2825745	0.2049387
15	1.7000060	0.0002142	0.0144842	78.2188369	0.2047619

component=N-HEPTANE

node	Z [m]	k	liqden [kg/m^3]	gasden [kg/m^3]	liqvis [kg/m hr]	gasvis [kg /m hr]
1	0.0000000	0.9616765	562.44116	3.4984885	0.7089690	0.0262425
2	0.1214290	0.9008409	608.60618	3.5056057	0.7222059	0.0260582
3	0.2428580	0.8916513	607.32075	3.4927826	0.7250542	0.0260193
4	0.3642870	0.8822504	606.00651	3.4811804	0.7279213	0.0259805
5	0.4857160	0.8725638	604.64927	3.4709295	0.7308196	0.0259414
6	0.6071450	0.8626170	603.25165	3.4620547	0.7337437	0.0259023
7	0.7285740	0.8524391	601.81660	3.4545792	0.7366877	0.0258632
8	0.8500030	0.8420621	600.34727	3.4485250	0.7396446	0.0258242
9	0.9714320	0.8315210	598.84702	3.4439127	0.7426064	0.0257854
10	1.0928610	0.8208526	597.31926	3.4407614	0.7455646	0.0257470
11	1.2142900	0.8100952	595.76743	3.4390887	0.7485098	0.0257089
12	1.3357190	0.7992881	594.19490	3.4389110	0.7514320	0.0256714
13	1.4571480	0.7884706	592.60490	3.4402429	0.7543208	0.0256346
14	1.5785770	0.7776815	591.00046	3.4430980	0.7571655	0.0255986
15	1.7000060	0.7666918	589.34383	3.4476171	0.7600241	0.0255626

node	Z [m]	liqdiff [m^2/hr]	gasdiff [m^2/hr]	kx [kmol/m^2hr]	ky [kmol/m^2 hr]
1	0.0000000	0.0002800	0.0135346	62.3148362	0.1744384
2	0.1214290	0.0002737	0.0134313	63.4910861	0.1752324
3	0.2428580	0.0002716	0.0134686	62.8341794	0.1752168
4	0.3642870	0.0002695	0.0135018	61.7501004	0.1751954
5	0.4857160	0.0002673	0.0135305	60.5846981	0.1751691
6	0.6071450	0.0002650	0.0135545	59.3404081	0.1751377
7	0.7285740	0.0002626	0.0135735	58.0200663	0.1751011
8	0.8500030	0.0002602	0.0135876	56.6273228	0.1750590
9	0.9714320	0.0002577	0.0135965	55.1665868	0.1750114
10	1.0928610	0.0002551	0.0136002	53.6429591	0.1749581
11	1.2142900	0.0002526	0.0135987	52.0621562	0.1748990
12	1.3357190	0.0002499	0.0135919	50.4304091	0.1748342
13	1.4571480	0.0002473	0.0135798	48.7543406	0.1747637
14	1.5785770	0.0002446	0.0135623	47.0408422	0.1746812
15	1.7000060	0.0002419	0.0135318	45.2878400	0.1745829

component=TOLUENE

node	Z [m]	k	liqden [kg/m^3]	gasden [kg/m^3]	liqvis [kg/m hr]	gasvis [kg /m hr]
1	0.0000000	0.7288963	749.99113	3.1900847	0.9549970	0.0296614
2	0.1214290	0.6829281	745.20136	3.1959985	0.9740892	0.0294531
3	0.2428580	0.6760761	744.84111	3.1843040	0.9782007	0.0294092
4	0.3642870	0.6690602	744.45025	3.1737129	0.9823404	0.0293652
5	0.4857160	0.6618244	744.01992	3.1643425	0.9865262	0.0293212
6	0.6071450	0.6543876	743.55094	3.1562154	0.9907507	0.0292770
7	0.7285740	0.6467716	743.04427	3.1493526	0.9950050	0.0292328
8	0.8500030	0.6390003	742.50096	3.1437743	0.9992791	0.0291887
9	0.9714320	0.6310998	741.92209	3.1394991	1.0035616	0.0291449
10	1.0928610	0.6230979	741.30877	3.1365444	1.0078400	0.0291015
11	1.2142900	0.6150233	740.66207	3.1349262	1.0121008	0.0290585
12	1.3357190	0.6069054	739.98300	3.1346594	1.0163295	0.0290162
13	1.4571480	0.5987739	739.27249	3.1357574	1.0205110	0.0289746
14	1.5785770	0.5906580	738.53132	3.1382324	1.0246298	0.0289339
15	1.7000060	0.5823853	737.74051	3.1422091	1.0287697	0.0288933

node	Z [m]	liqdiff [m^2/hr]	gasdiff [m^2/hr]	kx [kmol/m^2 hr]	ky [kmol/m^2 hr]
1	0.0000000	0.0005185	0.0144170	99.5431749	0.1954414
2	0.1214290	0.0005069	0.0143150	99.2493704	0.1964156
3	0.2428580	0.0005030	0.0143634	96.0905388	0.1964779
4	0.3642870	0.0004990	0.0144083	92.3534671	0.1965396
5	0.4857160	0.0004949	0.0144489	88.5618220	0.1966017
6	0.6071450	0.0004906	0.0144851	84.7265048	0.1966637
7	0.7285740	0.0004862	0.0145167	80.8585771	0.1967253
8	0.8500030	0.0004817	0.0145436	76.9697657	0.1967860
9	0.9714320	0.0004771	0.0145655	73.0723027	0.1968454
10	1.0928610	0.0004724	0.0145824	69.1787164	0.1969031
11	1.2142900	0.0004676	0.0145941	65.3016050	0.1969587
12	1.3357190	0.0004628	0.0146006	61.4534246	0.1970118
13	1.4571480	0.0004579	0.0146017	57.6462860	0.1970621
14	1.5785770	0.0004530	0.0145974	53.8917588	0.1971022
15	1.7000060	0.0004480	0.0145692	50.1909961	0.1970368

mixture properties

node	Z [m]	Temp [C]	pres [atm]	Liq.ax.Dis. [m^2/hr]	Vap.ax.Dis. [m^2/hr]
1	0.0000000	99.33048	1.02062354	0.0000646	0.26307
2	0.1214290	96.79564	1.01504046	0.0000638	0.26658
3	0.2428580	96.26072	1.00990666	0.0000648	0.26899
4	0.3642870	95.72594	1.00512049	0.0000653	0.26976
5	0.4857160	95.18905	1.00070973	0.0000658	0.27041
6	0.6071450	94.65109	0.99668264	0.0000664	0.27094
7	0.7285740	94.11320	0.99304740	0.0000670	0.27134
8	0.8500030	93.57671	0.98981207	0.0000676	0.27162
9	0.9714320	93.04301	0.98698457	0.0000682	0.27177
10	1.0928610	92.51363	0.98457264	0.0000689	0.27179
11	1.2142900	91.99017	0.98258381	0.0000696	0.27168
12	1.3357190	91.47430	0.98102534	0.0000703	0.27143
13	1.4571480	90.96770	0.97990421	0.0000710	0.27106
14	1.5785770	90.47211	0.97922702	0.0000717	0.27053
15	1.7000060	89.97732	0.97900000	0.0000725	0.26953

node	Z [m]	Liq.Enthalpy [kJ/kmol]	Vap.Enthalpy [kJ/kmol]
1	0.0000000	35305.0861	66740.8872
2	0.1214290	34764.5837	66268.3054
3	0.2428580	34592.6619	66099.9986
4	0.3642870	34414.8710	65925.0043
5	0.4857160	34229.9881	65742.9216
6	0.6071450	34038.1476	65553.9117
7	0.7285740	33839.5629	65358.2094
8	0.8500030	33634.5256	65156.1218
9	0.9714320	33423.4031	64948.0271
10	1.0928610	33206.6355	64734.3705
11	1.2142900	32984.7312	64515.6588
12	1.3357190	32758.2604	64292.4535
13	1.4571480	32527.8462	64065.3629
14	1.5785770	32294.1551	63835.0304
15	1.7000060	32055.5317	63708.5114

node	Z [m]	liqden [kg/m^3]	gasden [kg/m^3]	liqvis [kg /m hr]	gasvis [kg /m hr]
1	0.0000000	657.027708802	3.174531134	0.921013318	0.030512610
2	0.1214290	676.914577870	3.177624031	0.942251637	0.030338439
3	0.2428580	673.557426214	3.162900526	0.949522892	0.030311352
4	0.3642870	672.106911161	3.149066080	0.957116873	0.030285683
5	0.4857160	670.617740161	3.136233624	0.965098403	0.030261300
6	0.6071450	669.090241224	3.124422842	0.973474243	0.030238299
7	0.7285740	667.529016027	3.113653163	0.982248079	0.030216775
8	0.8500030	665.938897887	3.103943886	0.991420286	0.030196826
9	0.9714320	664.324867108	3.095314134	1.000987666	0.030178545
10	1.0928610	662.691951354	3.087782810	1.010943238	0.030162023
11	1.2142900	661.045116475	3.081368542	1.021276077	0.030147349
12	1.3357190	659.389168344	3.076089599	1.031971229	0.030134605
13	1.4571480	657.728665366	3.071963809	1.043009728	0.030123865
14	1.5785770	656.067834821	3.069008370	1.054368683	0.030115198
15	1.7000060	654.401879761	3.071102262	1.066144370	0.030088031

node	Z [m]	liqsurten [n/m]	intarea [m^2/m^3]	liqholdup [kmol]	gasholdup [kmol]
1	0.0000000	0.014984998	269.588759236	0.000041763	0.000000002
2	0.1214290	0.015232576	268.138515223	0.000043045	0.000000002
3	0.2428580	0.015261869	268.445643536	0.000042884	0.000000002
4	0.3642870	0.015292022	268.276852610	0.000042847	0.000000002
5	0.4857160	0.015323065	268.100792536	0.000042812	0.000000002
6	0.6071450	0.015354982	267.918193312	0.000042779	0.000000002
7	0.7285740	0.015387741	267.729133553	0.000042748	0.000000002
8	0.8500030	0.015421292	267.533786428	0.000042719	0.000000002
9	0.9714320	0.015455573	267.332419373	0.000042692	0.000000002
10	1.0928610	0.015490508	267.125393518	0.000042668	0.000000002
11	1.2142900	0.015526003	266.913163818	0.000042646	0.000000002
12	1.3357190	0.015561957	266.696275236	0.000042628	0.000000002
13	1.4571480	0.015598254	266.475353280	0.000042613	0.000000002
14	1.5785770	0.015634770	266.251095630	0.000042600	0.000000002
15	1.7000060	0.015672565	266.015244285	0.000042591	0.000000002

time= 0.8847001225
iteration no= 120

component=CYCLOHEXANE

node	Z [m]	k	liqden [kg/m^3]	gasden [kg/m^3]	liqvis [kg/m hr]	gasvis [kg /m hr]
1	0.0000000	1.9307209	676.74256	2.9297914	1.0385211	0.0314237
2	0.1214290	1.9125298	684.11050	2.9050179	1.0476664	0.0313516
3	0.2428580	1.9118620	682.75476	2.8787977	1.0523402	0.0313151
4	0.3642870	1.9108450	682.98676	2.8545143	1.0568342	0.0312802
5	0.4857160	1.9094432	683.18044	2.8322322	1.0611365	0.0312471
6	0.6071450	1.9076261	683.33273	2.8119986	1.0652372	0.0312157
7	0.7285740	1.9053609	683.44100	2.7938598	1.0691272	0.0311861
8	0.8500030	1.9026125	683.50250	2.7778612	1.0727991	0.0311582
9	0.9714320	1.8993432	683.51430	2.7640468	1.0762473	0.0311322
10	1.0928610	1.8955127	683.47329	2.7524596	1.0794679	0.0311081
11	1.2142900	1.8910777	683.37619	2.7431410	1.0824593	0.0310857
12	1.3357190	1.8859925	683.21948	2.7361309	1.0852223	0.0310652
13	1.4571480	1.8802083	682.99945	2.7314675	1.0877605	0.0310463
14	1.5785770	1.8736735	682.71212	2.7291875	1.0900800	0.0310292
15	1.7000060	1.8656081	682.36987	2.7294534	1.0923785	0.0310123

node	Z [m]	liqdiff [m^2/hr]	gasdiff [m^2/hr]	kx [kmol/m^2 hr]	ky [kmol/m^2 hr]
1	0.0000000	0.0012419	0.0153087	5.7541346	0.2744556
2	0.1214290	0.0012347	0.0153972	5.8348421	0.2747179
3	0.2428580	0.0012321	0.0155109	6.1279266	0.2747474
4	0.3642870	0.0012298	0.0156170	6.4611415	0.2747841
5	0.4857160	0.0012277	0.0157145	6.8457362	0.2748281
6	0.6071450	0.0012258	0.0158030	7.2892373	0.2748795
7	0.7285740	0.0012241	0.0158817	7.8002624	0.2749381
8	0.8500030	0.0012227	0.0159500	8.3886491	0.2750039
9	0.9714320	0.0012214	0.0160074	9.0655988	0.2750767
10	1.0928610	0.0012204	0.0160532	9.8438247	0.2751561
11	1.2142900	0.0012196	0.0160871	10.7377042	0.2752417
12	1.3357190	0.0012189	0.0161084	11.7634287	0.2753330
13	1.4571480	0.0012185	0.0161169	12.9391449	0.2754292
14	1.5785770	0.0012182	0.0161122	14.2850783	0.2755158
15	1.7000060	0.0012179	0.0160993	15.8196149	0.2755041

component=N-HEPTANE

node	Z [m]	k	liqden [kg/m^3]	gasden [kg/m^3]	liqvis [kg/m hr]	gasvis [kg /m hr]
1	0.0000000	1.2240312	591.26528	3.5388320	0.6591970	0.0269930
2	0.1214290	1.2092268	597.57632	3.5087235	0.6630701	0.0269311
3	0.2428580	1.2069879	596.46597	3.4767091	0.6650451	0.0268997
4	0.3642870	1.2046074	596.69788	3.4470676	0.6669414	0.0268698
5	0.4857160	1.2020668	596.89556	3.4198766	0.6687543	0.0268413
6	0.6071450	1.1993503	597.05637	3.3951928	0.6704800	0.0268143
7	0.7285740	1.1964405	597.17808	3.3730717	0.6721150	0.0267889
8	0.8500030	1.1933178	597.25832	3.3535681	0.6736565	0.0267650
9	0.9714320	1.1899605	597.29461	3.3367353	0.6751025	0.0267427
10	1.0928610	1.1863445	597.28433	3.3226250	0.6764517	0.0267219
11	1.2142900	1.1824434	597.22471	3.3112876	0.6777036	0.0267027
12	1.3357190	1.1782282	597.11284	3.3027716	0.6788590	0.0266850
13	1.4571480	1.1736672	596.94560	3.2971239	0.6799195	0.0266689
14	1.5785770	1.1687263	596.71971	3.2943895	0.6808879	0.0266541
15	1.7000060	1.1628509	596.44712	3.2947724	0.6818468	0.0266396

node	Z [m]	liqdiff [m^2/hr]	gasdiff [m^2/hr]	kx [kmol/m^2 hr]	ky [kmol/m^2 hr]
1	0.0000000	0.0000286	0.0133721	24.2422730	0.2238563
2	0.1214290	0.0000285	0.0134545	24.2819497	0.2241311
3	0.2428580	0.0000284	0.0135594	24.5929769	0.2242230
4	0.3642870	0.0000283	0.0136579	24.9002668	0.2243237
5	0.4857160	0.0000283	0.0137494	25.2221396	0.2244332
6	0.6071450	0.0000283	0.0138333	25.5586795	0.2245518
7	0.7285740	0.0000282	0.0139092	25.9098815	0.2246798
8	0.8500030	0.0000282	0.0139765	26.2756119	0.2248172
9	0.9714320	0.0000282	0.0140346	26.6555820	0.2249643
10	1.0928610	0.0000281	0.0140832	27.0493227	0.2251211
11	1.2142900	0.0000281	0.0141218	27.4561532	0.2252877
12	1.3357190	0.0000281	0.0141500	27.8751435	0.2254639
13	1.4571480	0.0000281	0.0141675	28.3050762	0.2256498
14	1.5785770	0.0000281	0.0141740	28.7444074	0.2258339
15	1.7000060	0.0000281	0.0141662	29.1843623	0.2258607

component=TOLUENE

node	Z [m]	k	liqden [kg/m^3]	gasden [kg/m^3]	liqvis [kg/m hr]	gasvis [kg /m hr]
1	0.0000000	0.9233033	771.91658	3.2285926	0.8834360	0.0305101
2	0.1214290	0.9121957	770.90776	3.2011886	0.8889914	0.0304400
3	0.2428580	0.9106669	771.13646	3.1721393	0.8918251	0.0304046
4	0.3642870	0.9090319	771.32800	3.1452392	0.8945465	0.0303707
5	0.4857160	0.9072766	771.47910	3.1205591	0.8971488	0.0303385
6	0.6071450	0.9053887	771.58698	3.0981507	0.8996262	0.0303079
7	0.7285740	0.9033547	771.64868	3.0780651	0.9019739	0.0302792
8	0.8500030	0.9011595	771.66105	3.0603519	0.9041878	0.0302521
9	0.9714320	0.8987863	771.62076	3.0450599	0.9062647	0.0302269
10	1.0928610	0.8962165	771.52421	3.0322362	0.9082029	0.0302034
11	1.2142900	0.8934299	771.36761	3.0219265	0.9100016	0.0301817
12	1.3357190	0.8904041	771.14688	3.0141748	0.9116618	0.0301617
13	1.4571480	0.8871150	770.85768	3.0090236	0.9131859	0.0301434
14	1.5785770	0.8835363	770.49537	3.0065134	0.9145777	0.0301267
15	1.7000060	0.8792569	770.00990	3.0068261	0.9159561	0.0301103

node	Z [m]	liqdiff [m^2/hr]	gasdiff [m^2/hr]	kx [kmol/m^2 hr]	ky [kmol/m^2 hr]
1	0.0000000	0.0000296	0.0134654	68.5128650	0.2415440
2	0.1214290	0.0000294	0.0135496	67.9940531	0.2418519
3	0.2428580	0.0000293	0.0136565	67.4278882	0.2419633
4	0.3642870	0.0000293	0.0137574	66.7746027	0.2420878
5	0.4857160	0.0000292	0.0138516	66.1066113	0.2422260
6	0.6071450	0.0000292	0.0139386	65.4223198	0.2423787
7	0.7285740	0.0000291	0.0140178	64.7199351	0.2425467
8	0.8500030	0.0000291	0.0140889	63.9974199	0.2427308
9	0.9714320	0.0000291	0.0141514	63.2524738	0.2429319
10	1.0928610	0.0000291	0.0142049	62.4825123	0.2431508
11	1.2142900	0.0000290	0.0142489	61.6846528	0.2433883
12	1.3357190	0.0000290	0.0142833	60.8557045	0.2436451
13	1.4571480	0.0000290	0.0143076	59.9921648	0.2439221
14	1.5785770	0.0000290	0.0143218	59.0902192	0.2442079
15	1.7000060	0.0000290	0.0143164	58.1312775	0.2442657

mixture properties

node	Z [m]	Temp [C]	pres [atm]	Liq.ax.Dis. [m^2/hr]	Vap.ax.Dis. [m^2/hr]
1	0.0000000	109.65134	1.06295039	0.0000602	0.58380
2	0.1214290	108.79958	1.05175186	0.0000604	0.59030
3	0.2428580	108.36860	1.04129034	0.0000606	0.59743
4	0.3642870	107.95691	1.03158211	0.0000608	0.60330
5	0.4857160	107.56524	1.02264895	0.0000609	0.60892
6	0.6071450	107.19415	1.01450618	0.0000611	0.61427
7	0.7285740	106.84410	1.00716863	0.0000612	0.61932
8	0.8500030	106.51543	1.00065048	0.0000614	0.62403
9	0.9714320	106.20833	0.99496529	0.0000615	0.62839
10	1.0928610	105.92283	0.99012581	0.0000617	0.63238
11	1.2142900	105.65879	0.98614401	0.0000618	0.63595
12	1.3357190	105.41588	0.98303094	0.0000620	0.63909
13	1.4571480	105.19356	0.98079668	0.0000622	0.64178
14	1.5785770	104.99107	0.97945035	0.0000623	0.64391
15	1.7000060	104.79106	0.97900000	0.0000625	0.64478

node	Z [m]	Liq.Enthalpy [kJ/kmol]	Vap.Enthalpy [kJ/kmol]
1	0.0000000	37626.8807	70889.7218
2	0.1214290	37470.3460	70789.2359
3	0.2428580	37432.8014	70754.4218
4	0.3642870	37400.6685	70722.9969
5	0.4857160	37374.1933	70695.0001
6	0.6071450	37353.4697	70670.4318
7	0.7285740	37338.5568	70649.2579
8	0.8500030	37329.4722	70631.4045
9	0.9714320	37326.1855	70616.7521
10	1.0928610	37328.6116	70605.1290
11	1.2142900	37336.6037	70596.3056
12	1.3357190	37349.9448	70589.9873
13	1.4571480	37368.3407	70585.8080
14	1.5785770	37391.4116	70583.3242
15	1.7000060	37415.3001	70561.7390

node	Z [m]	liqden [kg/m^3]	gasden [kg/m^3]	liqvis [kg /m hr]	gasvis [kg /m hr]
1	0.0000000	723.814616238	3.311146385	0.819743260	0.031150475
2	0.1214290	724.552754596	3.283767428	0.824531502	0.031076577
3	0.2428580	723.667061800	3.254676242	0.826226865	0.031032691
4	0.3642870	723.089854979	3.227795906	0.827789321	0.030989780
5	0.4857160	722.438143510	3.203198084	0.829212199	0.030947899
6	0.6071450	721.707706767	3.180934503	0.830492209	0.030907093
7	0.7285740	720.894263545	3.161055545	0.831627267	0.030867402
8	0.8500030	719.993333610	3.143609977	0.832616706	0.030828864
9	0.9714320	719.000234361	3.128644654	0.833461477	0.030791509
10	1.0928610	717.910070675	3.116204248	0.834164368	0.030755362
11	1.2142900	716.717732527	3.106330970	0.834730210	0.030720442
12	1.3357190	715.417900023	3.099064305	0.835166107	0.030686760
13	1.4571480	714.005050005	3.094440749	0.835481638	0.030654323
14	1.5785770	712.473464078	3.092493605	0.835689058	0.030623129
15	1.7000060	710.793752922	3.093037577	0.835905544	0.030603206

node	Z [m]	liqsurten [n/m]	intarea [m^2/m^3]	liqholdup [kmol]	gasholdup [kmol]
1	0.0000000	0.015505089	277.790319205	0.000045086	0.000000002
2	0.1214290	0.015582764	277.464213277	0.000045127	0.000000002
3	0.2428580	0.015586180	277.543954824	0.000045059	0.000000002
4	0.3642870	0.015585887	277.566100039	0.000045009	0.000000002
5	0.4857160	0.015581680	277.606589412	0.000044955	0.000000002
6	0.6071450	0.015573469	277.665812752	0.000044895	0.000000002
7	0.7285740	0.015561176	277.744089055	0.000044830	0.000000002
8	0.8500030	0.015544743	277.841643769	0.000044760	0.000000002
9	0.9714320	0.015524132	277.958596975	0.000044684	0.000000002
10	1.0928610	0.015499325	278.094952822	0.000044602	0.000000002
11	1.2142900	0.015470332	278.250588540	0.000044514	0.000000002
12	1.3357190	0.015437186	278.425241574	0.000044419	0.000000002
13	1.4571480	0.015399953	278.618499678	0.000044319	0.000000002
14	1.5785770	0.015358731	278.829791116	0.000044211	0.000000002
15	1.7000060	0.015315377	279.048329797	0.000044095	0.000000002

File= ERRORRES

I	J	Error (Temp.)	Error (x)	Error (xint)	Error (y)	Error (yint)	Error (L')	Error (V')
1	1	1.8764678232	0.9093554470	1.2564448953	0.1642178355	0.1322932971	0.3090183903	0.1444140275
1	2	3.9071657479	0.9526217176	1.4078734894	0.2240454064	0.2383654387	0.1019735848	0.1402629977
1	3	2.5294159302	1.3734047745	0.4612212968	0.1429811189	0.1468046921	0.0354806223	0.0953709819
1	4	0.9850845237	0.3647155858	0.5297422932	0.0550149973	0.0553241414	0.0101388935	0.0407127904
1	5	0.1678900274	0.4022802048	0.0546065389	0.0128199575	0.0125794419	0.0118195207	0.0107833071
1	6	0.1487688468	0.0327736761	0.0446952159	0.0108964102	0.0116124393	0.0062879129	0.0067758622
1	7	0.1165075995	0.0186740225	0.0366725800	0.0075264937	0.0077958903	0.0025836337	0.0047150866
1	8	0.0485049631	0.0415275526	0.0065942074	0.0030539232	0.0031193385	0.0002284979	0.0019802508
1	9	0.0117181276	0.0031097630	0.0095838230	0.0007139298	0.0006956243	0.0003760052	0.0005406743
1	10	0.0061128520	0.0068673129	0.0016558919	0.0003980935	0.0004153719	0.0003343127	0.0002587466
1	11	0.0075034548	0.0005193227	0.0008530947	0.0004630543	0.0004866409	0.0002286991	0.0002928746
1	12	0.0050089270	0.0008061287	0.0011471076	0.0003044846	0.0003162368	0.0000929117	0.0001949993
1	13	0.0021993304	0.0010059248	0.0005851998	0.0001320808	0.0001350673	0.0000172143	0.0000875174
1	14	0.0005772706	0.0006042498	0.0002323916	0.0000343660	0.0000338024	0.0000122668	0.0000253081
1	15	0.0001490942	0.0002660596	0.0000544134	0.0000105169	0.0000109903	0.0000100166	0.0000074834
1	16	0.0001494147	0.0000807253	0.0000103435	0.0000097144	0.0000102546	0.0000042773	0.0000059712
1	17	0.0000875790	0.0000139022	0.0000113275	0.0000057789	0.0000059113	0.0000007108	0.0000035719
70	1	0.0001617579	0.0800395615	0.0835338442	0.0000722183	0.0000054387	0.0062800481	0.0000059937
70	2	0.2339949803	0.0994890998	2.4103580048	0.0105834383	0.0098783617	0.0198021388	0.0126677912
70	3	0.2780421154	2.0619589543	1.2431307665	0.0112222342	0.0115014964	0.0065135733	0.0114376513
70	4	0.0967378106	1.2208593115	0.8656830075	0.0036942820	0.0045779756	0.0027616153	0.0035189666
70	5	0.0882713377	0.8503345326	0.6392260341	0.0035798335	0.0035715047	0.0031113060	0.0038439794

70	6	0.0405623204	0.6158916495	0.4646268372	0.0013716093	0.0014670447	0.0020093476	0.0013301261
70	7	0.0349504064	0.4522021831	0.3537210447	0.0013109233	0.0013717104	0.0013716663	0.0013864408
70	8	0.0258269522	0.3391016044	0.2638297714	0.0008987627	0.0009164529	0.0010919107	0.0008988332
70	9	0.0187591506	0.2546660909	0.1991804392	0.0006568726	0.0006840548	0.0007797855	0.0006893980
70	10	0.0145216538	0.1914051439	0.1496151881	0.0004975947	0.0005122410	0.0005962619	0.0005128805
70	11	0.0033453069	0.0332676648	0.0554163766	0.0000949280	0.0001246164	0.0000777950	0.0000793951
70	12	0.0029990307	0.0236135760	0.0540311732	0.0000549161	0.0000587797	0.0000527914	0.0000537170
70	13	0.0026930184	0.0249384410	0.0365341589	0.0000815087	0.0000785663	0.0000716162	0.0000849055
70	14	0.0014718565	0.0145654424	0.0278401142	0.0000313979	0.0000397335	0.0000255416	0.0000287759
70	15	0.0014474981	0.0122303031	0.0203039699	0.0000410020	0.0000403361	0.0000326803	0.0000429174
70	16	0.0008268237	0.0085028951	0.0150383840	0.0000160964	0.0000176587	0.0000189948	0.0000162093
70	17	0.0006858936	0.0064525581	0.0110826724	0.0000166172	0.0000168308	0.0000147616	0.0000176162
70	18	0.0004887627	0.0046963921	0.0081805092	0.0000107731	0.0000109402	0.0000113113	0.0000109881
70	19	0.0003558428	0.0034903195	0.0060320482	0.0000080301	0.0000082620	0.0000078211	0.0000084108
70	20	0.0002698159	0.0025633673	0.0044516116	0.0000060978	0.0000061801	0.0000060693	0.0000063062
70	21	0.0001034309	0.0004813132	0.0021913419	0.0000015413	0.0000017834	0.0000009604	0.0000014154
70	22	0.0001234742	0.0006956387	0.0021661433	0.0000025330	0.0000025248	0.0000015900	0.0000026128
70	23	0.0000737656	0.0005363880	0.0013153893	0.0000015204	0.0000015032	0.0000012343	0.0000015908
120	1	0.3856078526	0.2252035789	2.4719901312	0.0251734335	0.0234323498	0.0511820703	0.0318451927
120	2	0.6701268201	1.5504957706	1.6147207606	0.0440043961	0.0427767946	0.0561515703	0.0506495558
120	3	0.4761728986	1.0673744206	0.7477099997	0.0291997251	0.0320561658	0.0106065181	0.0328039285
120	4	0.2271424769	0.6369369845	0.0276111271	0.0144143183	0.0150358936	0.0099809033	0.0168159804
120	5	0.1501044616	0.0571044943	0.0303872379	0.0079588236	0.0084673079	0.0038463710	0.0097123587
120	6	0.0904710671	0.0428663260	0.0742023755	0.0044320470	0.0046416113	0.0029450416	0.0057082253
120	7	0.0577825949	0.0610728636	0.0089204631	0.0024955904	0.0026732381	0.0014201560	0.0034373428
120	8	0.0366539323	0.0102661503	0.0245787608	0.0014881969	0.0015715928	0.0010219961	0.0021622388

120	9	0.0237399780	0.0193908050	0.0055359669	0.0008944300	0.0009598315	0.0005868878	0.0013772126
120	10	0.0154365097	0.0051678612	0.0079521712	0.0005625277	0.0005996268	0.0004042091	0.0008994832
120	11	0.0152739484	0.0051812240	0.0057010690	0.0005448214	0.0005838307	0.0003900646	0.0008913866
120	12	0.0117759338	0.0046711119	0.0048431852	0.0004142263	0.0004441429	0.0003026695	0.0006905359
120	13	0.0083538956	0.0041242252	0.0030935074	0.0002914410	0.0003129313	0.0002149290	0.0004934528
120	14	0.0057408937	0.0030573700	0.0021910260	0.0001996343	0.0002143543	0.0001490744	0.0003417429
120	15	0.0038936778	0.0022079292	0.0014192054	0.0001353518	0.0001454135	0.0001016307	0.0002334795
120	16	0.0026237137	0.0015108919	0.0009477552	0.0000912567	0.0000981207	0.0000687107	0.0001582187
120	17	0.0017731400	0.0010048913	0.0006650741	0.0000619428	0.0000664582	0.0000470863	0.0001077332
120	18	0.0011933304	0.0007035500	0.0004222461	0.0000416474	0.0000448267	0.0000315079	0.0000725988
120	19	0.0008051807	0.0004523957	0.0002785040	0.0000282276	0.0000303145	0.0000214138	0.0000492848
120	20	0.0005498317	0.0002990558	0.0002288063	0.0000193723	0.0000207250	0.0000149678	0.0000338083
120	21	0.0004944601	0.0002299999	0.0001770206	0.0000173568	0.0000186588	0.0000132334	0.0000303412
120	22	0.0004002525	0.0001858506	0.0001407398	0.0000140521	0.0000151218	0.0000106356	0.0000245865
120	23	0.0003063782	0.0001545604	0.0001131871	0.0000107870	0.0000115977	0.0000082153	0.0000188813
120	24	0.0002230188	0.0001336466	0.0000684816	0.0000078026	0.0000084580	0.0000057926	0.0000136792
120	25	0.0001611943	0.0000982414	0.0000588677	0.0000057193	0.0000061295	0.0000043675	0.0000100165
120	26	0.0001178416	0.0000811938	0.0000447070	0.0000041860	0.0000044706	0.0000032649	0.0000073077
120	27	0.0000812465	0.0000692009	0.0000119100	0.0000028121	0.0000030772	0.0000020075	0.0000049123

File= PRODUCTRES

time [min]= 1.32705
time iteration step= 3
corrector step= 14
product taken=CYCLOHEXANE
reflux ratio (L/D)= 2.33333
distillate rate [kmol/h]= 0.0594711790
amount of product in the collector [kmol]= 0.0013002764
product composition in the collector= 0.6718990712
reflux rate [kmol/h]= 0.138766
reflux composition= 0.6649992663
boilup rate [kmol/h]= 0.191778
pressure drop [mm Water]= 10.14596
flooding pressure [mm Water]=255.43643

reboiler mixture

still temperature [c]= 94.731346077

component	x,mol%	yboil,mol%
CYCLOHEXANE	0.35908492	0.51757818
N-HEPTANE	0.28639596	0.24885431
TOLUENE	0.35451912	0.23356751

rebhold [kmol]= 0.095795148
conhold [kmol]= 0.000565891
tot. liq. hold. in packed section [kmol]= 0.000598656
tot. gas. hold. in packed section [kmol]= 0.000000028
qreb [kj/hr]= 6144.766430039
qcond [kj/hr]= 6186.836332674

time [min]= 26.54100
time iteration step= 60
corrector step= 59
product taken=CYCLOHEXANE
reflux ratio (L/D)= 2.33333
distillate rate [kmol/h]= 0.0978790140
amount of product in the collector [kmol]= 0.0345642236
product composition in the collector= 0.6009948470
reflux rate [kmol/h]= 0.228384
reflux composition= 0.5190724197
boilup rate [kmol/h]= 0.315736
pressure drop [mm Water]= 33.08173
flooding pressure [mm Water]=255.43643

reboiler mixture

still temperature [c]= 99.330407054

component	x,mol%	yboil,mol%
CYCLOHEXANE	0.23469636	0.36873075
N-HEPTANE	0.31533010	0.30339629
TOLUENE	0.44997354	0.32787296

rebhold [kmol]= 0.062535654
conhold [kmol]= 0.000562731
tot. liq. hold. in packed section [kmol]= 0.000597363
tot. gas. hold. in packed section [kmol]= 0.000000028
qreb [kj/hr]= 10270.609686137
qcond [kj/hr]= 10322.717855366

time [min]= 46.44676
time iteration step= 105
corrector step= 24
product taken=CYCLOHEXANE
reflux ratio (L/D)= 2.33333
distillate rate [kmol/h]= 0.1249338127
amount of product in the collector [kmol]= 0.0717031930
product composition in the collector= 0.4794542993
reflux rate [kmol/h]= 0.291512
reflux composition= 0.1724527882
boilup rate [kmol/h]= 0.409624
pressure drop [mm Water]= 59.85459
flooding pressure [mm Water]=255.43643

reboiler mixture

still temperature [c]= 106.783707041

component	x,mol%	yboil,mol%
CYCLOHEXANE	0.05387914	0.09806604
N-HEPTANE	0.30447967	0.34797197
TOLUENE	0.64164120	0.55396199

rebhold [kmol]= 0.025393852
conhold [kmol]= 0.000561106
tot. liq. hold. in packed section [kmol]= 0.000601821
tot. gas. hold. in packed section [kmol]= 0.000000028
qreb [kj/hr]= 13527.854362004
qcond [kj/hr]= 13587.708703508

Appendix 6. Thermocouple Calibration Curves

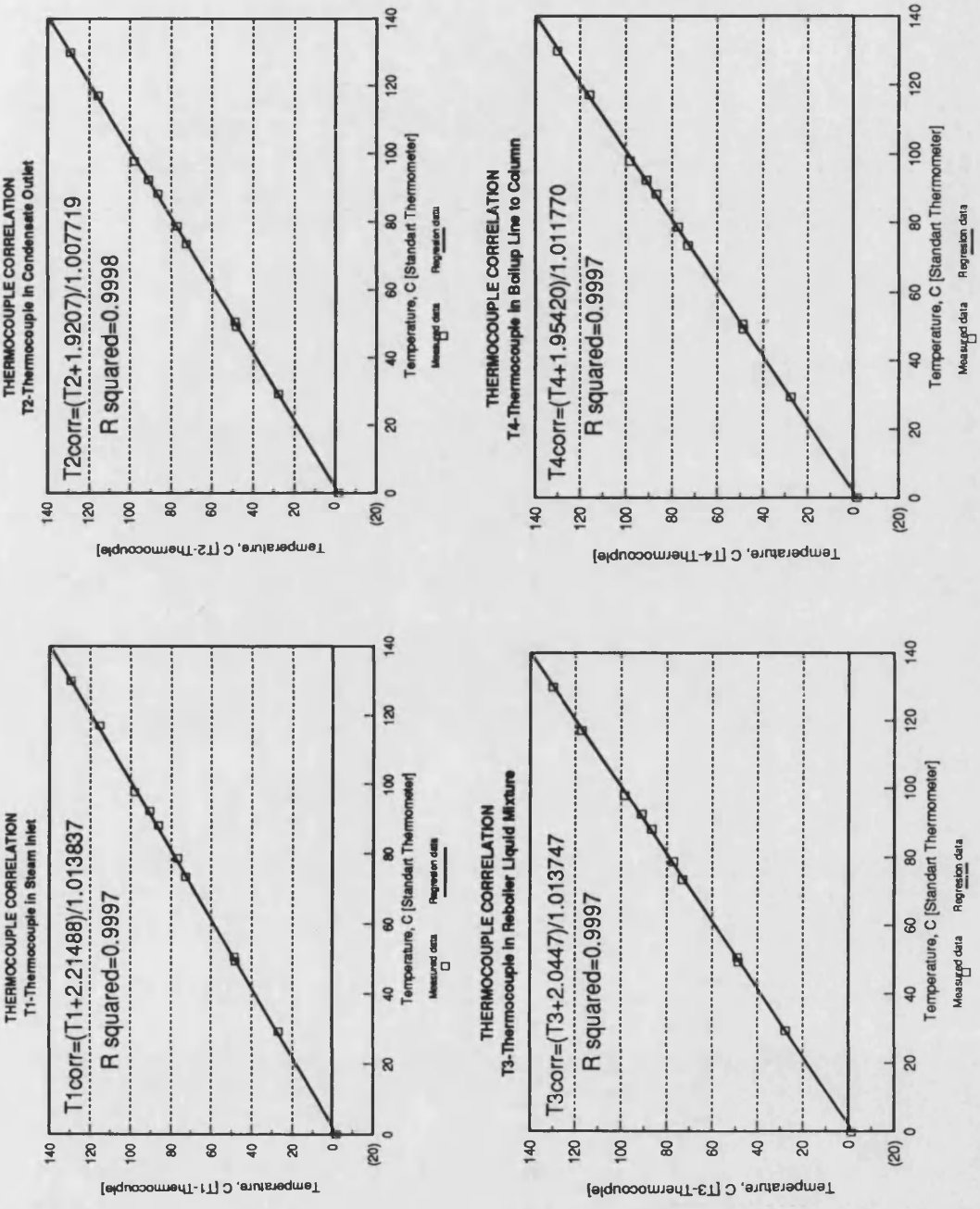


Figure A6.1 Calibration Curves for T1, T2, T3 and T4.

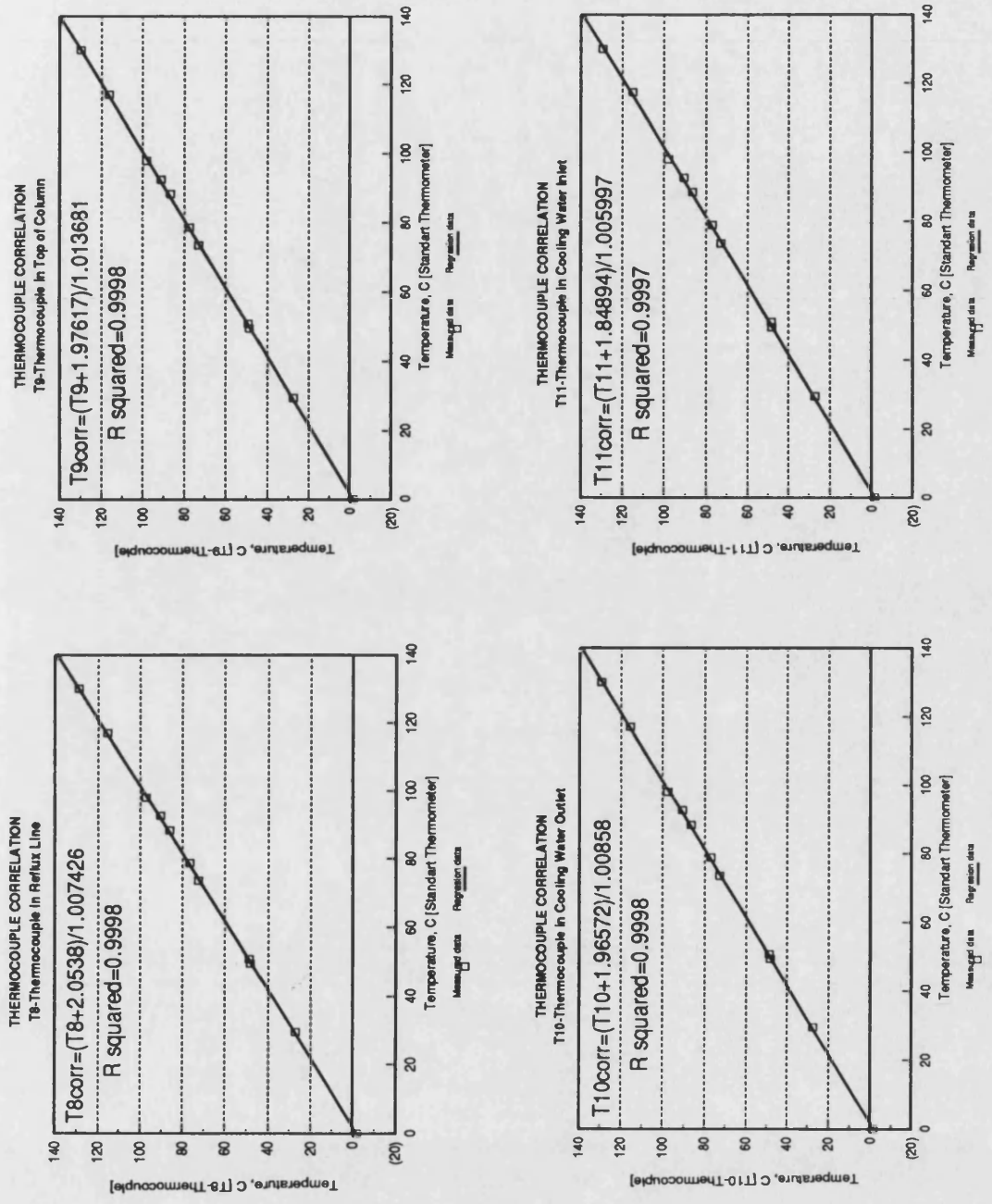


Figure A6.2 Calibration Curves for T8, T9, T10 and T11.

Appendix 7. The Listing of the Control Program "DISTCONT"

```
 10 MODE 131
 20 REM TITLE "DISTCONT"
 30 REM Control Program for Packed Batch Distillation Experiments
 40 REM It switches the controller between constant reflux ratio
 45 REM and constant overhead composition
 50 REM It takes temperature and pressure readings every 5 seconds
 55 REM It actuates the valves
 60 ON ERROR GOTO 3470
 70 PROCsetup
 80 REPEAT
 90 realtime%=TIME DIV 100 : PROCscan : PROCdisplay
   : PROCloop
 95 PROCsetflags : PROCdoflags : PROCwipemessage
100 timestep%=5
110 T%=T%+500
120 REM T%=T%+timestep%*100
130 REPEAT : PROCinput : UNTIL TIME > T%
140 UNTIL FALSE
150 END
160
170 DEF PROCsetup
180 *FX5,1
190 *FX6,0
200 *FX7,7
210 *FX8,7
220 REM disable spooling
230 *FX3,16
240 TIME=0 : T%=0 : U%=0 : V%=0
250 CLS
260 fileflag%=FALSE : heatflag=FALSE : loopflag=FALSE
265 filename$="" : saveperiod%=0 : setpoint = 0 :setxtop= 0.9
266 blank$=STRING$(79," ") : messagecounter%=0 : realtime%=0
267 M%=0 : input$="" : command$=""
270 reflux%=100 : samplelength%=4 : samplingflag%=FALSE
275 samplingflagb%=FALSE :samplingflagt%=FALSE
   : sampleperiod%=0
276 writeperiod%=0 : sampleno%=0:KC=264:Ti=146
   :xaver=0:NO%=1
```

```
280 saveflag%=FALSE : writeflag%=FALSE : tsampleflag%=FALSE
285 bsampleflag%=FALSE : gsampleflag%=FALSE:piconflag%=FALSE
290 DIM analogs(16) : DIM labels$(12): DIM constant(11)
  : DIM coeff(11)
300 RESTORE 340
  : FOR N%=1 TO 11 : READ labels$(N%) : NEXT
310 RESTORE 350
  : FOR N%=1 TO 18:READ A$:PRINT TAB(55,N%);A$ : NEXT
320 RESTORE 390 : FOR D%=1 TO 11
330 READ constant(D%), coeff(D%) : NEXT
340 DATA STEAM_INLET,CONDENSATE_OUT,MIXLIQ_REBOILER
345 DATA BOILUP_REBOILER PACKING(42.5CM),PACKING(85CM)
346 DATA PACKING(127.5CM),LIQ_REFLUX
347 DATA TOP_COLUMN, COOLING_WATER_OUT,
    COOLING_WATER_IN
350 DATA R-set reflux ratio,S-set sampling time,O-open disc file
355 DATA C-close disc file,P-pressure set point,H-heat on/off
356 DATA L-control loop on/off,D-disc write period
360 DATA T-take top sample,B-take bottom sample,
    G-take both samples
365 DATA F-set sample period,W-set print period
366 DATA I-Comp.Cont. Loop on/off,E-Conc. set point
367 DATA K-Set proportional gain
368 DATA M-Set integral time cons.,N-Set comp. (C:1/H:2/T:3)
370 RESTORE 390 : FOR D%=1 TO 11
380 READ constant(D%), coeff(D%) : NEXT
390 DATA 2.21488,1.013837,1.9207,1.007719,2.0447,1.013747,1.9542
395 DATA 1.01177,0.0,1.0,0.0,1.0,0.0,1.0,2.0538,1.007426,1.97617
396 DATA 1.013681,1.96572,1.00858,1.84894,1.005997
400 ENDPROC
410
420
430 DEF PROCscan
440 LOCAL T%,D%
450 FOR D%=1 TO 16
460 T%=TIME
470 T%=T%+5
480 REM T%=T%+timestep%
490 REPEAT : UNTIL TIME>T%
500 analogs(D%)=FNanalog_in(D%)
```

```
510 NEXT : ENDPROC
520
530 DEF FNanalog_in(A%)
540 REM =12.7 : ENDPROC : REM ******
550 LOCAL A$,B$
560 IF (A%<1) OR (A%>16) THEN ="" : ENDPROC
570 B$=RIGHT$("00"+STR$~(A%),2)
580 *FX2,1
590 *FX3,19
600 *FX15,1
610 PRINT CHR$(2)+B$;
620 FOR N%=0 TO 100 : NEXT
630 *FX3,18
640 INPUT LINE A$
650 REM *FX15,1
660 FOR N%=0 TO 1 : NEXT
670 *FX3,16
680 *FX2,0
690 =VAL(MID$(A$,4))
700 ENDPROC
710
720 DEF PROCvalve(chan%,val)
730 LOCAL B$
740 IF (chan%<1) OR (chan%>4) THEN ENDPROC
750 IF (val<0) THEN val=0
760 IF (val>1) THEN val=1
770 mask%=2^(chan%-1):mask%=mask%*65536
  :IF val=1 THEN M%=(M% OR mask%)
780 IF val=0 THEN M%=M% AND NOT mask%
790 B$=RIGHT$(("000000"+STR$~(M%)),6)
800 PROCout("14 "+B$)
810 ENDPROC
820
830 DEF PROCpulse(chan%,time%)
840 PROCvalve(chan%,1)
850 time%=TIME + time%
860 REPEAT : UNTIL TIME > time%
870 PROCvalve(chan%,0)
880 ENDPROC
890
```

```
900 DEF PROCrefcontrol
910 LOCAL B$,A$,r%
920 IF NO%=1 THEN xaver=xtopcyc
930 IF NO%=2 THEN xaver=xtophep
940 IF NO%=3 THEN xaver=xtoptol
980 @%=&020105 : PRINTTAB(0,22);"TCOMP=";xaver
990 error=setxtop-xaver
1000 inttimes%=0
1010 errortop=errortop+error
1015 REM PI CONTROLLER
1020 reflux%=80+KC*error+KC*timestep%/Ti*errortop
1025 REM PID CONTROLLER
1030 REM reflux%=80+KC*error+KC*timestep%/Ti*errortop
     +KC*TD/timestep%*(error-errold)
1040 REM OSCLI("*FX3,10"):PROCprintit(reflux%):OSCLI("*FX3,16")
1050 r%=100-reflux% : IF r%<0 THEN r%=0
1060 IF r%>100 THEN r%=100
1070 B$="100"+RIGHT$("00"+STR$(r%),3)
1080 PROCout("13 "+B$)
1090 ENDPROC
1100 REM
1110 DEF PROCsetreflux
1120 LOCAL B$,A$,r%
1130 reflux%=VAL(MID$(command$,2))
1140 REM OSCLI("*FX3,10"):PROCprintit(reflux%):OSCLI("*FX3,16")
1150 REM IF reflux%<>100 then TIME=0:T%=0
1160 r%=100-reflux% : IF r%<0 THEN r%=0
1170 IF r%>100 THEN r%=100
1180 B$="100"+RIGHT$("00"+STR$(r%),3)
1190 PROCout("13 "+B$)
1200 ENDPROC
1210
1220 DEF PROCpicontrol
1230 IF MID$(command$,2,1)="0" AND piconflag% THEN
     piconflag%=FALSE : reflux%=100
1240 IF MID$(command$,2,1)="1" AND NOT piconflag% THEN
     piconflag%=TRUE
1250 inttimes%=0:errortop=0
1260 ENDPROC
1270 DEF PROCinput
```

```
1280 LOCAL A$,B$,C$
1290 A$=INKEY$(1):IF A$=CHR$(13) THEN command$=input$
   :input$="": PROCexecute(command$) : ENDPROC
1300 input$=input$+A$ : ENDPROC
1310
1320 DEF PROCexecute(command$)
1330 *FX3,16
1340 LOCAL A$,A%
1350 PRINT TAB(0,21);"command is ";command$ : REM ENDPROC
1360 A$=LEFT$(command$,1)
   : A%=INSTR("RrSsOoCcPpHhLlDdTtFfWwBbGgIiEeKkMmNn",A$)
   : IF A%=0 THEN PROCmessage("no such command!")
   : ENDPROC
1370 A%=(A%+1) DIV 2
1380 ON A% GOSUB 1410,1420,1430,1440,1450,1460,1470,1480,1510,1530,
          1540,1520,1500,1550,1490,1560,1570,1580
1390 ENDPROC
1400
1410 PROCsetreflux : RETURN
1420 PROCsetsample : RETURN
1430 PROCopfile : RETURN
1440 PROCclfile : RETURN
1450 PROCsetpress : RETURN
1460 PROCswitchheat : RETURN
1470 PROCswitchloop : RETURN
1480 PROCsetdisc : RETURN
1490 PROCsettopconcen :RETURN
1500 PROCtakesample : RETURN
1510 PROCtakesampletop : RETURN
1520 PROCtakesamplebot : RETURN
1530 PROCfrequsample : RETURN
1540 PROCsetwrites : RETURN
1550 PROCpicontrol:RETURN
1560 PROCsetgain :RETURN
1570 PROCsettimecons : RETURN
1580 PROCsetcomp : RETURN
1590
1600 DEF PROCsetgain : KC=VAL(MID$(command$,2))
     : PROCmessage("KC"+STR$(KC)) : ENDPROC
1610 DEF PROCsetcomp : NO%=VAL(MID$(command$,2))
```

```
    : PROCmessage("NO"+STR$(NO%)) : ENDPROC
1620 DEF PROCsettimecons : Ti=VAL(MID$(command$,2))
    : PROCmessage("Ti"+STR$(Ti)) : ENDPROC
1630 DEF PROCsettopconcen : setxtop=VAL(MID$(command$,2))
    : PROCmessage("setxtop"+STR$(setxtop)) : ENDPROC
1640 DEF PROCdisplay
1650 *FX3,16
1660 LOCAL N%,X$ : X$="    "
1670 PRINT TAB(0,1);"TEMPERATURES [C]  MT  CT"
1680 @%=&020105 : FOR N%=1 TO 11
1690 PRINT TAB(0,N%+1);labels$(N%);TAB(20);X$;
  TAB(17,N%+1),analogs(N%)
1700 analogs(N%)=(constant(N%)+analogs(N%))/coeff(N%)
    : PRINTTAB(23,N%+1);analogs(N%) : NEXT
1710 REM for CYCLOHEXANE-TOLUENE BINARY SYSTEM
1720 REM reflux line liquid sample concentration
1730 CONCAN=0.96013815-0.0192278*(analogs(9)-80)
      +0.00224945*(analogs(9)-80)^2
      -0.0000948*(analogs(9)-80)^3
1740 REM set the model equation given below
1750 REM xtopcyc=-1.588009+0.055573*analogs(9)-0.0004714*(analogs(9)^2)
        +(1.19128*FNabspres*0.9869)
1760 REM for CYCLOHEXANE-N.HEPTANE BINARY SYSTEM
1770 REM set the model equation given below
1780 REM xtopcyc=7.294534-0.14198*analogs(9)
        +0.000501*(analogs(9)^2)
        +(1.904607*FNabspres*0.9869)
1790 REM for TERNARY SYSTEM (Cyclohexane-n.Heptane-Toluene)
1815 xtopcyc=-7.23609+0.213447*analogs(9)-0.00139*(analogs(9)^2)
1816 xtophep=3.372894-0.09715*analogs(9)+0.000693*(analogs(9)^2)
1821 xtoptol=4.858739-0.11619*analogs(9)+0.0007*(analogs(9)^2)
1830 @%=&020205 : PRINTTAB(0,16);"Diffpress      ";
  TAB(21,16);"[mm Water] ";TAB(10,16),FNdiffpres
1840 @%=&020305 : PRINTTAB(0,17);"Abspress      ";
  TAB(21,17);"[bar] ";TAB(10,17),FNabspres
1850 @%=&020408 : PRINTTAB(0,18);"Comp. Mol Frac.=";
  xtopcyc;"  ";xtophep;"  ";xtoptol
1860 @%=&90A : REM general format?
1870 PRINTTAB(30,1);"Elapsed time   ";TAB(45,1);realtime%
1880 PRINTTAB(30,2);"Filename       ";TAB(45,2);filename$
```

```
1890 PRINTTAB(30,3);"Save every      ";TAB(45,3)saveperiod%
1900 PRINTTAB(30,4);"Sample each     ";TAB(45,4);sampleperiod%
1910 PRINTTAB(30,5);"Writing each    ";TAB(45,5);writeperiod%
1920 PRINTTAB(30,6);"Setpointpres    ";TAB(45,6);setpoint
1930 PRINTTAB(30,7);"Setpointxtop    ";TAB(45,7);setxtop
1940 @%=&020104
   : PRINTTAB(30,8);"Sample time     ";TAB(45,8);
   samplelength%/10 : @%=&90A
1950 PRINTTAB(30,9);"Reflux          ";TAB(45,9);reflux%
1960 PRINTTAB(30,10);"Steam          ";TAB(45,10);FNbool(heatflag)
1970 PRINTTAB(30,11);"Control heat   ";TAB(45,11);FNbool(loopflag)
1980 PRINTTAB(30,12);"Control conc   ";TAB(45,12);FNbool(piconflag%)
1990 PRINTTAB(30,13);"Gain           ";TAB(45,13);KC
2000 PRINTTAB(30,14);"Time Constant  ";TAB(45,14);Ti
2010 PRINTTAB(30,15);"Comp.          ";TAB(45,15);NO%
2020 ENDPROC
2030
2040 DEF PROCloop
2050 IF NOT loopflag THEN ENDPROC
2060 IF (((analogs(13)-1.0)/4.0*20) > setpoint) AND heatflag THEN
      PROCvalve(4,0) : heatflag=FALSE : ENDPROC
2070 IF (((analogs(13)-1.0)/4.0*20) < setpoint)
     AND NOT heatflag THEN PROCvalve(4,1) : heatflag=TRUE
2080 ENDPROC
2090
2100 DEF PROCmessage(A$)
2110 *FX3,16
2120 PRINTTAB(0,24);blank$;
2130 PRINTTAB(5,24);CHR$(136);"Message: ";A$;TAB(0,0)
2140 messagecounter%=2 : VDU7 : ENDPROC
2150
2160 DEF PROCsetsample : LOCAL A%
2170 A%=VAL(MID$(command$,2))*10 : REM tenths of a second
2180 IF A%<0 THEN A%=0
2190 IF A%>100 THEN A%=100 : REM 0 to ten secs.
2200 samplelength%=A% : ENDPROC
2210
2220 DEF PROCtakesample : samplingflag%=TRUE : ENDPROC
2230 DEF PROCtakesamplebot : samplingflagb%=TRUE : ENDPROC
2240 DEF PROCtakesampletop : samplingflagt%=TRUE : ENDPROC
```

```
2250
2260
2270 DEF PROCsamplenow : LOCAL B$
   : B$=RIGHT$("000"+STR$(samplelength%),3)
2280 PROCout("11 "+B$+B$):REM SAMPLE FROM THE REBOILER
2290 B$=RIGHT$("000"+STR$(samplelength%),3)
2300 PROCout("12 "+B$+B$)
   : REM SAMPLE FROM THE REFLUX LINE
2310 ENDPROC
2320
2330 DEF PROCsamplenowbot
   : LOCAL B$ : B$=RIGHT$("000"+STR$(samplelength%),3)
2340 PROCout("11 "+B$+B$)
   : REM SAMPLE FROM THE REBOILER
2350 ENDPROC
2360 DEF PROCsamplenowtop
   : LOCAL B$ : B$=RIGHT$("000"+STR$(samplelength%),3)
2370 PROCout("12 "+B$+B$)
   : REM SAMPLE FROM THE REFLUX LINE
2380 ENDPROC
2390 DEF PROCopfile
   : IF LEN(command$) > 1 AND LEN(command$) <= 10 THEN
     filename$=MID$(command$,2)
2400 OSCLI("*SPOOL "+filename$)
2410 PROCmessage("Opening "+filename$) : ENDPROC
2420
2430 DEF PROCclfile
2440 IF filename$="" THEN PROCmessage("File not open!")
   : ENDPROC
2450 PROCmessage("Close file?? (Y/N)")
2460 REPEAT: A$=GET$: A% =INSTR("YyNn",A$)
   : UNTIL A%<>0
2470 IF (A%+1) DIV 2 =1 THEN OSCLI("*SPOOL")
   : PROCmessage("File closed")
   :filename$="" :message$="" : fileflag%=FALSE
   : saveflag%=FALSE : ENDPROC
2480 PROCmessage("Make your mind up!")
2490 ENDPROC
2500
2510 DEF PROCsetpress : setpoint=VAL(MID$(command$,2))
```

```
   : PROCmessage("setpoint "+STR$(setpoint)) : ENDPROC
2520
2530 DEF PROCswitchheat
   : IF MID$(command$,2,1)="1" AND NOT heatflag THEN
     PROCvalve(4,1) : heatflag=TRUE
2540 IF MID$(command$,2,1)="0" AND heatflag THEN PROCvalve(4,0)
   : heatflag=FALSE
2550 ENDPROC
2560
2570 DEF PROCswitchloop : IF MID$(command$,2,1)="0" THEN
                    loopflag = FALSE : ENDPROC
2580 IF MID$(command$,2,1)="1" THEN loopflag = TRUE
   : ENDPROC
2590 PROCmessage("0 or 1, silly!"):ENDPROC
2600
2610 DEF PROCsetdisc
   : REM IF filename$="" THEN
     PROCmessage("Open file first!") : REM ENDPROC
2620 saveperiod%=VAL(MID$(command$,2))
   : IF saveperiod%<>0 THEN U%=realtime%
2630 ENDPROC
2640
2650
2660 DEF PROCsetwrites : writeperiod%=VAL(MID$(command$,2))
   : IF saveperiod%<>0 THEN W%=realtime%
2670 ENDPROC
2680 DEF PROCwipemessage
2690 IF (messagecounter%=0)THEN ENDPROC
2700 messagecounter%=messagecounter%-1
   : IF (messagecounter%=0) THEN
     PRINTTAB(0,24);blank$;
2710 ENDPROC
2720
2730 DEF PROCout(B$)
2740 *FX3,16
2750 PRINTTAB(0,19);"Output ";B$;"    "
2760 REM ENDPROC : REM ******
2770 B$=CHR$(1)+B$
2780 *fx2,1
2790 *FX3,19
```

```
2800 *FX15,1
2810 PRINT B$
2820 FOR N%=0 TO 100 : NEXT
2830 REPEAT:UNTIL GET=10 : REM INPUT LINE B$
2840 *FX3,16
2850 *FX2,0
2860 ENDPROC
2870
2880 DEF FNbool(a)
2890 IF a THEN ="ON " : ENDPROC
2900 ="OFF" : ENDPROC
2910
2920 DEF PROCfrequsample
   : sampleperiod%=VAL(MID$(command$,2))
   : IF sampleperiod% <> 0 THEN V% = realtime%
2930 ENDPROC
2940
2950 DEF FNdiffpres
2960 LOCAL A : A=(analogs(13)-1.0)/4.0*20: =A : ENDPROC
2970
2980 DEF FNabspres:LOCAL A:A=(analogs(14) - 1)/4*2 : =A
   : ENDPROC
2990
3000 DEF PROCprint(Z%)
3010 LOCAL N%,A
3020 VDU15 : REM select 12 to the inch
3030 @%=&020006
3040 PRINT realtime%,;
3050 @%=&020106
   : FOR N% = 1 TO 11 : PRINT analogs(N%),;:NEXT
3060 @%=&020207 : PRINT FNdiffpres,;
3070 @%=&020306 : PRINT FNabspres,;
3080 @%=&020306 : PRINT xtopcyc,;
3090 PRINT" reflux = "; : @%=&02003 : PRINT reflux%,;
3100 IF Z%<>0 THEN PRINT" sample no. ";
            : @%=&02003:PRINT Z%,;
3110 PRINT
3120 @%=&90A
3130 ENDPROC
3140
```

```
3150 DEF PROCprintit(K%)
3160 LOCAL N%,A
3170 VDU15 : REM select 12 to the inch
3180 @%=&020006
3190 PRINT realtime%,;
3200 @%=&020106
   : FOR N% = 1 TO 11 : PRINT analogs(N%),;:NEXT
3210 @%=&020207 : PRINT FNdiffpres,;
3220 @%=&020306 : PRINT FNabspres,;
3230 @%=&020306 : PRINT xtopcyc,;
3240 IF K%<>0 THEN PRINT" reflux ratio ";
   : @%=&020003:PRINT K%,;
3250 PRINT
3260 @%=&90A
3270 ENDPROC
3280 DEF PROCsetflags
3290 IF (writeperiod% <> 0) THEN IF W% <= realtime% THEN
     writeflag% = TRUE : W%=W%+writeperiod%
3300 IF (sampleperiod% <> 0) THEN IF V% <= realtime% THEN
     samplingflag% = TRUE : V%=V%+sampleperiod%
3310 IF (saveperiod% <> 0 AND filename$ <> "") THEN
     IF U% <= realtime% THEN fileflag% = TRUE
     : U%=U%+saveperiod%
3320 ENDPROC
3330
3340 DEF PROCdoflags : LOCAL message$,devicemask%
   : devicemask%=18
3350 IF samplingflag% THEN PROCsamplenow : OSCLI("*FX3,10")
   : sampleno%=sampleno%+1 : PROCprint(sampleno%)
   : OSCLI("*FX3,16") : PROCmessage("Sampling now!")
   : PROCclearflags : ENDPROC
3360 IF samplingflagt% THEN PROCsamplenowtop : OSCLI("*FX3,10")
   : sampleno%=sampleno%+1 : PROCprint(sampleno%)
   : OSCLI("*FX3,16")
   : PROCmessage("Sampling now!") : PROCclearflags : ENDPROC
3370 IF samplingflagb% THEN PROCsamplenowbot : OSCLI("*FX3,10")
   : sampleno%=sampleno%+1 : PROCprint(sampleno%)
   : OSCLI("*FX3,16") : PROCmessage("Sampling now!")
   : PROCclearflags : ENDPROC
3380 IF fileflag% THEN devicemask%=devicemask% AND NOT 16
```

```
   : message$=message$+"Saving " : REM enable spooling
3390 IF writeflag% THEN devicemask%=devicemask% OR 8
   : message$=message$+"writing" : REM enable printer
3400 OSCLI("*FX3,"+STR$(devicemask%)) : PROCprint(0)
   : OSCLI("*FX3,16") : PROCclearflags
3410 IF message$ <>"" THEN PROCmessage(message$)
3420 IF piconflag% THEN PROCrefcontrol
3430 ENDPROC
3440
3450 DEF PROCclearflags : writeflag% = FALSE
   : samplingflag% =FALSE :samplingflagb% =FALSE
   :samplingflagt% =FALSE : fileflag% = FALSE
   : ENDPROC
3460
3470 REM error handler
3480 *fx3,16
3490 IF ERR = 17 THEN ON ERROR OFF : command$="R100"
   : PROCsetreflux:END
3500 PRINTTAB(0,24);blank$;
3510 PRINTTAB(0,23); : REPORT : PRINT" E R R O R";
   : messagecounter%=2 : SOUND 1,-15,40,40
   : FILENAME$="": GOTO 80
```

Appendix 8. Refractometer Analysis for the Binary Mixture

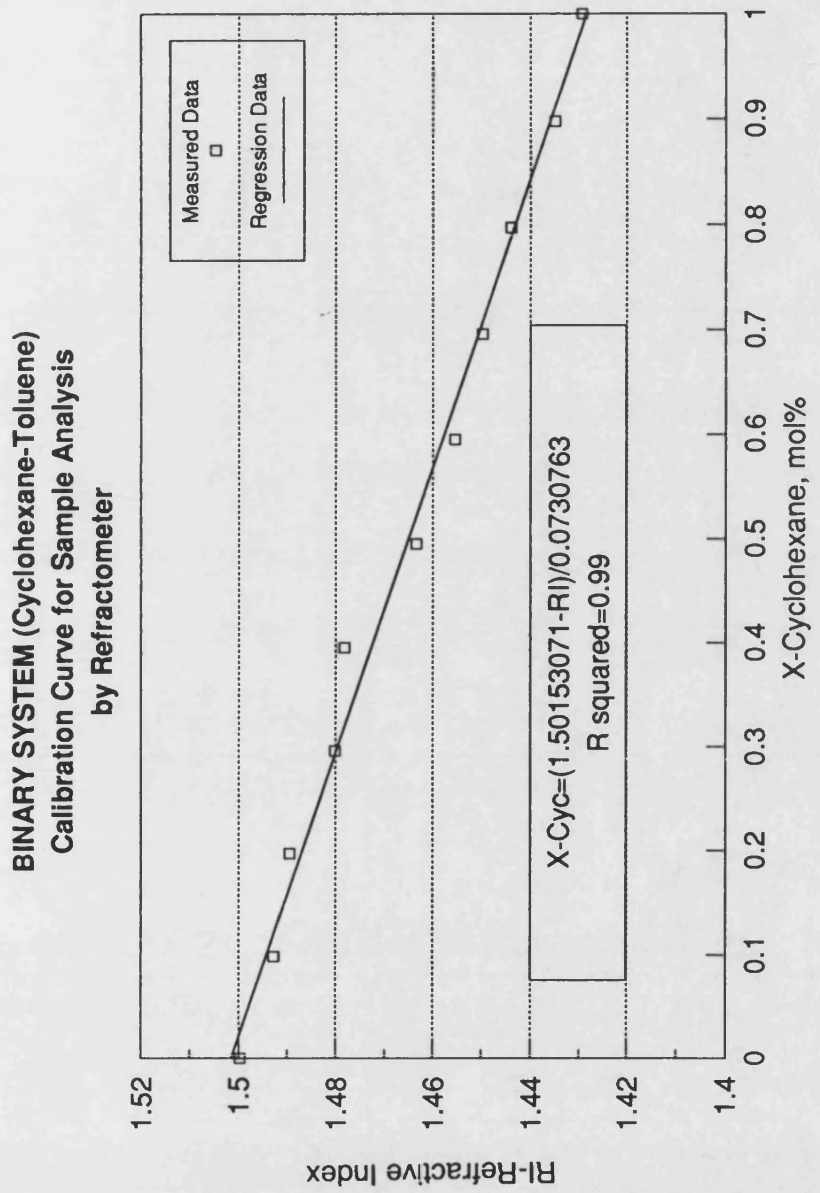
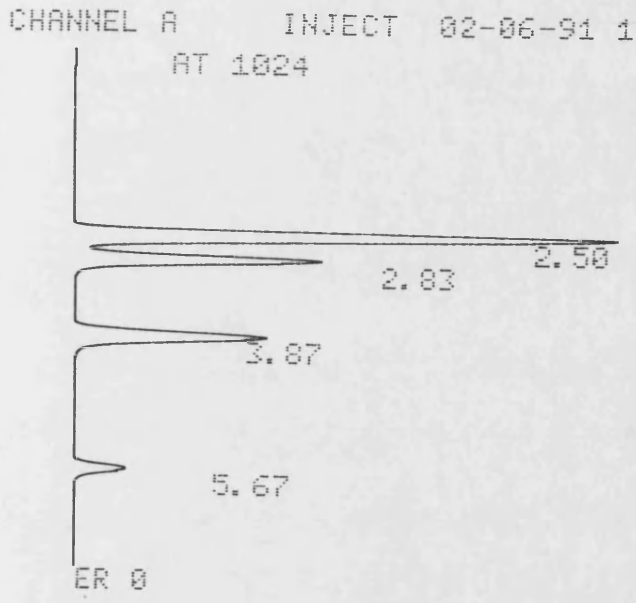


Figure A8 Correlation between Refractive Index and Cyclohexane Mol Fraction of the Binary System (Cyclohexane-Toluene).

Appendix 9. Gas Chromotography Analysis for the Ternary Mixture



CALIB 02-06-91 13:36:24

FILE 1. METHOD 2. RUN 17 INDEX 1

ANALYST: SERAP

NAME		MOL%	RT	AREA BC	RF
CYLHEX		17.483	2.5	6292171 02	1.66
HEPTAN		14.658	2.83	2962593 03	2.956
TOLUEN		47.646	3.87	2716481 01	10.479
4	INTERNAL STD		5.67	597422 01	
TOTALS		79.787		12568667	

Figure A9.1 A Typical Plotter Output of Gas Chromotography Analysis

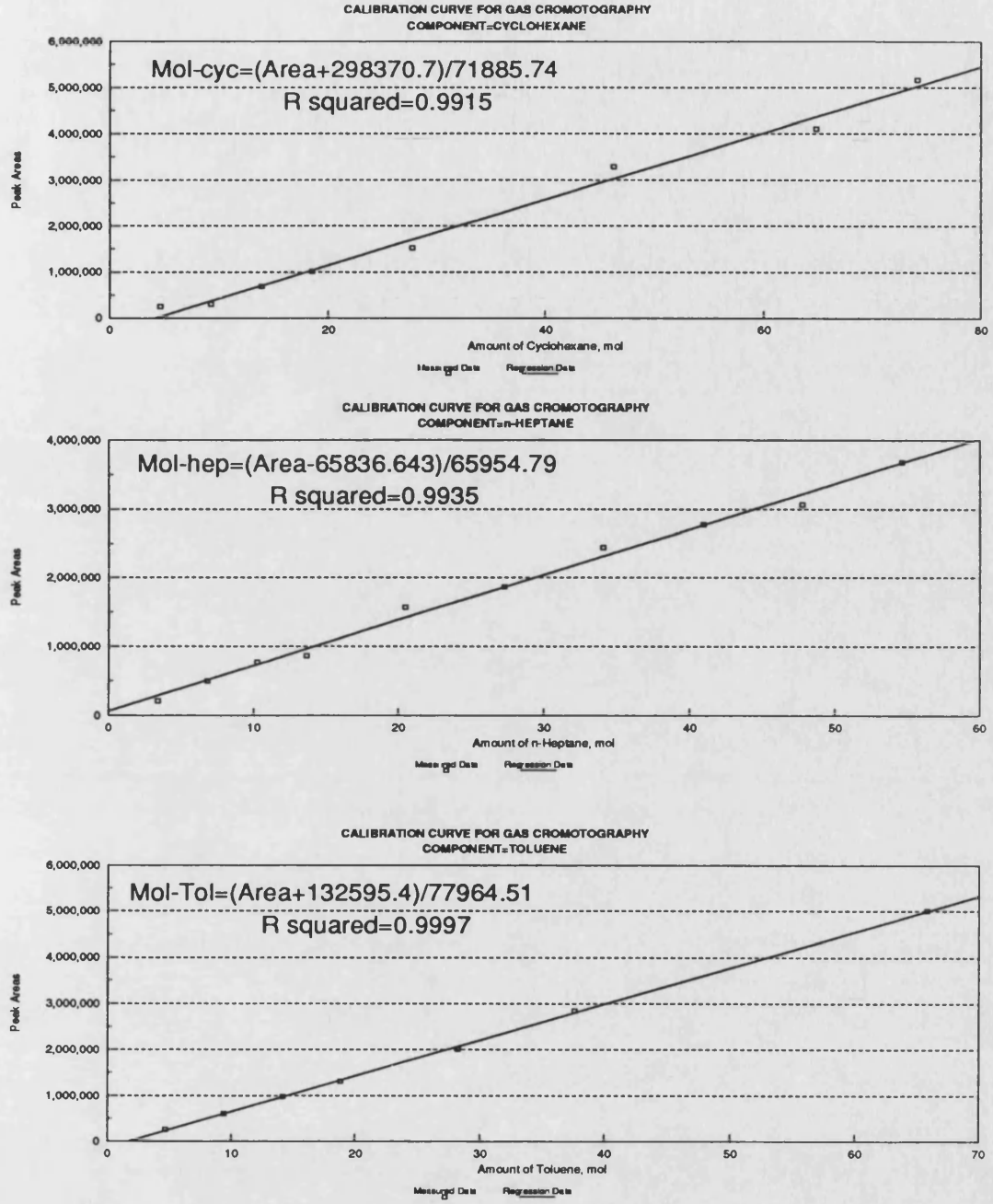


Figure A9.2 Correlations between Peak Areas and Mol Fractions of Cyclohexane, n.Heptane and Toluene.

Appendix 10. Estimated Composition Curves for the Ternary System

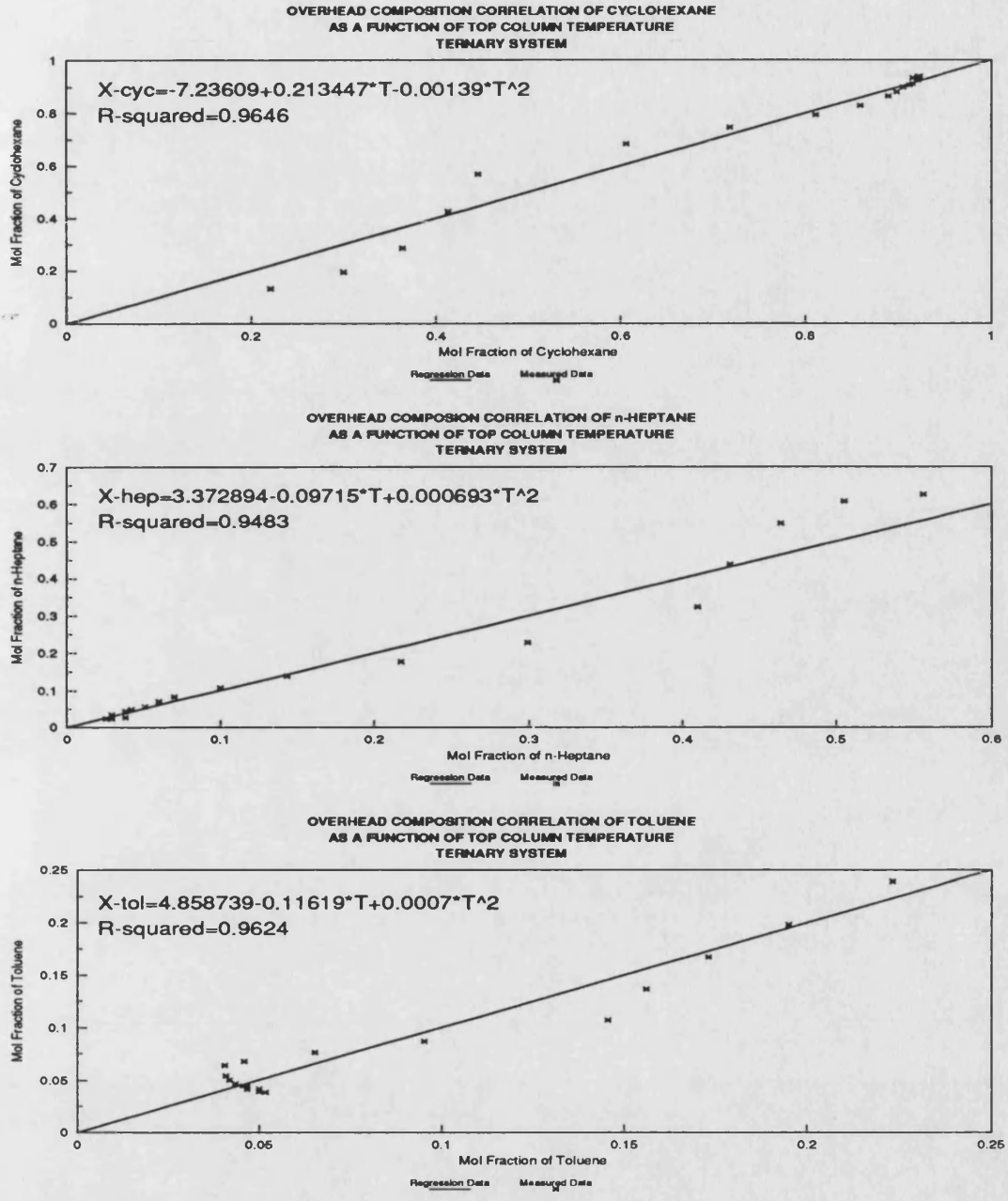


Figure A10 Estimated Mol Fractions of Cyclohexane, n.Heptane and Toluene Curves as Functions of Top Column Temperature and Pressure.

Appendix 11. The Listing of the Simulation Package, "PACBACDIS"

Main Fortran 77 File:	**pacbacdis.f** It includes routines for packed batch distillation simulation
Include Files	**comblok.f** It includes COMMON BLOCKS of pacbacdis.f **param.f** It defines the dimensions of the problem and it should be modified for each problem at hand. nmc : the number of components of the reboiler charge numg: the number of finite difference nodes taken in the packed section
Input Data Files	**binary.dat** for the binary mixtures **ternary.dat** for the ternary mixtures or multicomponent mixtures They should be modified according to the problem specifications
DASSL Files	**ddassl.f** **daux.f** **dlinal.f**

The programs should be compiled to produce an executable file (simul.out) as follows:

f77 pacbacdis.f ddassl.f dlinal.f daux.f -o simul.out

The listing of the programs developed in this work PACBACDIS.F, COMBLOK.F and PARAM.F are given below.

```
c*********************************************************************
c*      PACBACDIS.F                                                  *
c*      simulation program for packed batch distillation column      *
c*      multicomponent, multipurpose with holdup                     *
c*      solution of partial differential equations by finite         *
c*      difference method and dassl                                  *
c*********************************************************************

      implicit double precision (a-z)
      external fcn,dres1,dres2,dres3,dres4,dres5
      integer numg,lrw,liw,idid,ipar,info(15),morvol,prog,artis,order
      integer nmc,ii,jj,comp,compon,n,typack,yaztot,yazpro,neq,simstop
      integer j,kk,l,i,iter,cmp,iteres,option,solstart,decide,model
      integer simpro,opmode,objfonk,var,kindex,pacmat,iwork(250)
      integer stopcr
      include 'param.f'
      include 'comblok.f'

      common/previous/totlhold,totghold,time,temppre,xmolpre,
     +   xintpre,ymolpre,yintpre,prxref,
     +   xrefpre,xfpre,xfprime
      common/dec3/qbinit,qbartis,qbmax,qc,relaxgiven

      dimension temppre(1:numg),xintpre(1:nmc,1:numg)
      dimension prxref(1:nmc),sol(2*numg),yintpre(1:nmc,1:numg)
      dimension v(1:nmc),yprime(2*numg)
      dimension solen(2*numg),elapsed(1:nmc)
      dimension xfprime(1:nmc),xmolpre(1:nmc,1:numg),xrefpre(1:nmc)
      dimension rwork(42000),ymolpre(1:nmc,1:numg),xfpre(1:nmc)
      dimension xpurity(1:nmc),reflux(1:nmc)
      dimension solprime(2*numg),switch(1:20),varref(1:20)


      character *15 namecomp(1:nmc)
      character *85 aaaa
      character *20 molresult
      character *20 flowresult
      character *20 phyresult
      character *20 errorres
      character *20 totalref
      character *20 productres
      character *20 moltotal
      character *20 flowtotal
      character *20 phytotal
      character *20 errortotal
      character *20 modelres

c ********************************************************
c     definition of the system
      artis=0
c ********************************************************

c
c     take all input data from input.dat file
c
c     coulson, chem. eng., vol. 6, page 765
c     sherwood, the properties of gases and liquids, 3.ed, page 629
c     chemcad data sources

      if(nmc.eq.2) then
         open (12,file='binary.dat')
      else
         open (12,file='ternary.dat')
      end if

      read(12,*)aaaa,aaaa,aaaa,aaaa,height,aaaa,coldia
      read(12,*)aaaa,aaaa,aaaa,aaaa,typack
      read(12,*)aaaa,aaaa,specarea,aaaa,packfact,aaaa,voidfrac
      read(12,*)aaaa,aaaa,pacmat,aaaa,dp,aaaa,presbot
      read(12,*)aaaa,prestop,aaaa,aaaa,aaaa,numcomp
      read(12,*)aaaa
      read(12,*)(namecomp(i),i=1,nmc)
      read(12,*)aaaa,charge,aaaa
      read(12,*)(xfeed(i),i=1,nmc)
      read(12,*)aaaa,(a1(i),i=1,nmc)
      read(12,*)aaaa
      read(12,*)(b1(i),i=1,nmc)
      read(12,*)aaaa
      read(12,*)(c1(i),i=1,nmc)
      read(12,*)aaaa
      read(12,*)(d1(i),i=1,nmc)
      read(12,*)aaaa
      read(12,*)(e1(i),i=1,nmc)
      read(12,*)aaaa
      read(12,*)(wme(i),i=1,nmc)
      read(12,*)aaaa
      read(12,*)(tbp(i),i=1,nmc)
      read(12,*)aaaa
      read(12,*)(w(i),i=1,nmc)
      read(12,*)aaaa
      read(12,*)(tc(i),i=1,nmc)
      read(12,*)aaaa
      read(12,*)(pc(i),i=1,nmc)
      read(12,*)aaaa
      read(12,*)(vc(i),i=1,nmc)
      read(12,*)aaaa
      read(12,*)(solpar(i),i=1,nmc)
      read(12,*)aaaa
      read(12,*)(tden(i),i=1,nmc)
      read(12,*)aaaa
      read(12,*)(lden(i),i=1,nmc)
      read(12,*)aaaa
      read(12,*)(visa(i),i=1,nmc)
      read(12,*)aaaa
      read(12,*)(visb(i),i=1,nmc)
      read(12,*)aaaa
      read(12,*)(cpa(i),i=1,nmc)
      read(12,*)aaaa
      read(12,*)(cpb(i),i=1,nmc)
      read(12,*)aaaa
      read(12,*)(cpc(i),i=1,nmc)
      read(12,*)aaaa
      read(12,*)(cpd(i),i=1,nmc)
      read(12,*)aaaa,aaaa,aaaa,aaaa,aaaa,aaaa,decide
      if(decide.eq.1) then
      read(12,*)aaaa,aaaa,vboil
      end if
      if(decide.eq.2) then
      read(12,*)aaaa,aaaa,qreb
      end if
      if(decide.eq.3) then
      read(12,*)aaaa,aaaa,qbinit,qbartis,qbmax
      end if
      read(12,*)aaaa,volrdrum,aaaa,volconhold
      read(12,*)aaaa,aaaa,aaaa,aaaa,option,aaaa,aaaa,vollhold
      read(12,*)aaaa,volghold,aaaa,aaaa,aaaa,aaaa,aaaa,opmode,aaaa
      read(12,*)aaaa,(reflux(i),i=1,nmc)
      read(12,*)aaaa,var,aaaa
      read(12,*)(switch(i),varref(i),i=1,var)
      read(12,*)aaaa,aaaa,(xpurity(i),i=1,nmc)
      read(12,*)aaaa,aaaa,kcgain,aaaa,timecons
      read(12,*)aaaa,aaaa,aaaa,aaaa,aaaa,objfonk
      read(12,*)aaaa,aaaa,aaaa,model
      read(12,*)aaaa,aaaa,aaaa,deltaz,aaaa,relaxfac
      read(12,*)aaaa,yaztot,aaaa,yazpro
      read(12,*)aaaa,molresult,moltotal,aaaa,flowresult,flowtotal
      read(12,*)aaaa,phyresult,phytotal
      read(12,*)aaaa,errorres,errortotal,aaaa,totalref,aaaa,productres
      read(12,*)aaaa,modelres
      read(12,*)aaaa,aaaa,aaaa,solstart
      read(12,*)aaaa,alfaky,aaaa,alfakx,aaaa,beta1,aaaa,beta2
      read(12,*)aaaa,aaaa,aaaa,simpro,aaaa,simstop,aaaa,errnorm
      read(12,*)aaaa,aaaa,aaaa,stopcr,aaaa
      read(12,*)(elapsed(i),i=1,nmc)
      if(pacmat.eq.1) then
         pacsurten=75.0d-3
```

```
      end if
      if(pacmat.eq.2) then
        pacsurten=61.0d-3
      end if
      if(pacmat.eq.3) then
         pacsurten=33.0d-3
      end if
      if(pacmat.eq.4) then
         pacsurten=56.0d-3
      end if
      relaxgiven=relaxfac
C     For the stability of the solution
c     r=deltatime/(deltaz)^2 <=0.5
c     deltatime=0.4d0*(deltaz)**2.0d0
      deltatime=60.0d0/3600.0d0

c  ****************************************************************
      n=2*numg
c     number of nodes considered in the packed section is given by
c     numg= (height of the column / spatial step increment)+1

c     numg=height/deltaz+1

c     equivalent diameter of the packing [m]
      eqdia=6.0d0*(1.0d0-voidfrac)/specarea

c       cross area of the column [m2]
      crosarea=3.1416d0*coldia**2.0d0/4.0d0
      crosarea=crosarea*voidfrac
c       volume of the column [m3]
      colvol=crosarea*height

c       quantity [kmol] and composition [molar] of the feed charge
c       feed is at its bubble point
      rebhold(1)=charge


      deltapres=(presbot-prestop)/(numg-1)
      do 65 ii=1,numg
^Ip(ii)=presbot-(ii-1)*deltapres
 65   continue

      call initialise

c     ***********************************************
      time=0.0d0
      totdistil=0.0d0
      distil=0.0d0
c     ***********************************************
      open(14,file=molresult)
      open(15,file=phyresult)
      open(16,file=flowresult)
      open(18,file=errorres)
      open(20,file=productres)
      open(30,file=modelres)
c       ----------------------------------------------------------------
c       startup procedure - total reflux conditions
c       if solstart=1 then do the calculations
c       else read the values from totalref file
      if(solstart.eq.1) then
        call start(qreb)
      else
        open(19,file=totalref)
c       read total reflux calculation results from totalref file
        read(19,7113)tstill,rebhold(1),conhold,qreb,qc,vboil
        read(19,7117)(xfeed(jj),jj=1,nmc)
        read(19,7117)(yboil(jj),jj=1,nmc)
        read(19,6789)((xmolfrac(i,j),j=1,numg),i=1,nmc)
        read(19,6789)((ymolfrac(i,j),j=1,numg),i=1,nmc)
        read(19,6789)((xint(i,j),j=1,numg),i=1,nmc)
        read(19,6789)((yint(i,j),j=1,numg),i=1,nmc)
        read(19,6789)(ttk(j),j=1,numg)
        read(19,6789)(liqrate(j),j=1,numg)
        read(19,6789)(gasrate(j),j=1,numg)
      end if

 7113   format(6(d22.16,1x))
 7111   format(8(d22.16,1x))
 7117   format(3(d22.16,1x))
 6789   format(30(d22.16,1x))
      if(simstop.eq.1) then
         goto 4000
      end if
c       ----------------------------------------------------------------
      prog=2


      preconhold=conhold
      do 781 i=1,nmc
        xfprime(i)=0.0d0
 781  continue
      do 77 i=1,nmc
        prxref(i)=0.0d0
 77   continue
      call propcal

c     equimolar counterdiffusion total Ni=0


c       asign old xpre and ypre values
      do 103 cmp=1,nmc
        xref(cmp)=xmolfrac(cmp,numg)
        xrefpre(cmp)=xref(cmp)
        xfpre(cmp)=xfeed(cmp)
        do 102 ii=1,numg
          xpre(cmp,ii)=xmolfrac(cmp,ii)
          ypre(cmp,ii)=ymolfrac(cmp,ii)
 102    continue
 103  continue
      do 106 ii=1,numg
        blpre(ii)=liqhold(ii)
        bgpre(ii)=gashold(ii)
        mixhlpre(ii)=mixhl(ii)
        mixhvpre(ii)=mixhv(ii)
 106  continue

      conhlpre=mixhl(numg)
c     ******************************************************
c       step 1
      iter=1
      iteres=1
c     initial values from the total reflux condition calculations
c     reboiler inial values
c     but retain gas rates and all other initial values from
c     the total reflux calculations
      morvol=1
c       ----------------------------------------------------------------
c       opmode=3 for constant overhead composition
c       keep xref(morvol)=xpurity(morvol)
c       check if d(xref(morvol))/dt=0 for the given reflux ratio(L/D)
c       if not try with a new reflux ratio
      stepref=2.0d0
      if(opmode.eq.3) then
        open(22,file='savedat')
c       write  reflux calculation results  to 'savedat' file
        write(22,7111)tstill,rebhold(1),conhold,qreb,qc,vboil,
     +      totdistil,distil
        write(22,7117)(xfeed(jj),jj=1,nmc)
        write(22,7117)(yboil(jj),jj=1,nmc)
        write(22,6789)((xmolfrac(i,j),j=1,numg),i=1,nmc)
        write(22,6789)((ymolfrac(i,j),j=1,numg),i=1,nmc)
        write(22,6789)((xint(i,j),j=1,numg),i=1,nmc)
        write(22,6789)((yint(i,j),j=1,numg),i=1,nmc)
        write(22,6789)(ttk(j),j=1,numg)
        write(22,6789)(liqrate(j),j=1,numg)
        write(22,6789)(gasrate(j),j=1,numg)
```

```
          close (unit=22)
            err2=xpurity(morvol)-xmolfrac(morvol,numg)
            ref2=98.0d0
        end if




c  *********************************************************
c                          step 2

c  decide=1 then constant boilup rate is given vboil [kmol/hr]
c  decide=2 then qreb [kj/hr] is given and vboil will be calculated.
c  decide=3 then qreb and vboil both will be calculated


c  separation of components from the mixture

       order=1
       do 178 morvol=1,nmc


         if(morvol.gt.1) then
           opmode=1
         end if
       xtotdistil=0.0d0
c  desired purity of the product is restricted
c  by total reflux results
         write(*,55544)reflux(morvol)
55544   format('dynamic simulation',/,'reflux ratio=L/D=',f10.5)
c  quantity of more volatile component in distil

2000  if(opmode.eq.3) then
3001      call dynamic(errnorm,morvol,distil,reflux,iter,iteres,
     +      xtotdistil,totdistil)
          err1=xpurity(morvol)-xmolfrac(morvol,numg)
          ref1=reflux(morvol)
          if(dabs(err1).gt.dabs(err2)) then
              reflux(morvol)=ref2
              ref1=ref2
              success1=ref2
              goto 3009
          end if
          success1=ref1
          if(dabs(err1).le.errnorm.or.reflux(morvol).ge.99.0d0) then
               goto 3006
          end if
3003      reflux(morvol)=ref1+stepref
          ref2=reflux(morvol)
          call dynamic(errnorm,morvol,distil,reflux,iter,iteres,
     +      xtotdistil,totdistil)
          err2=xpurity(morvol)-xmolfrac(morvol,numg)
          if(dabs(err2).le.errnorm.or.reflux(morvol).ge.99.0d0) then
               goto 3006
          end if
          if(dabs(err2).lt.dabs(err1)) then
               success2=ref2
          end if
          reflux(morvol)=ref1-stepref
          ref3=reflux(morvol)
          if(ref3.le.0.0d0) then
             err3=5.0d0
             goto 666
          end if
          call dynamic(errnorm,morvol,distil,reflux,iter,iteres,
     +      xtotdistil,totdistil)
          err3=xpurity(morvol)-xmolfrac(morvol,numg)
666       if(dabs(err3).le.errnorm.or.reflux(morvol).ge.99.0d0) then
               goto 3006


          end if
          if(dabs(err3).lt.dabs(err2)) then
               success2=ref3
               err2=err3
               ref2=ref3
               goto 3008
          end if
          if(dabs(err1).lt.dabs(err2).and.dabs(err1).lt.dabs(err3)) then
c              ref1 is the successful point, reduce the step
3009           stepref=stepref/2.0d0
               reflux(morvol)=ref1
               success1=ref1
               goto 3003
          end if
c              ref2 is the successful point, accelaration step
3008           reflux(morvol)=2.0d0*success2-success1
               if(reflux(morvol).lt.0.0d0) then
                  reflux(morvol)=success2
               end if
               ref1=reflux(morvol)
               success1=ref2
               goto 3001


       end if

       if(opmode.eq.1) then
         call dynamic(errnorm,morvol,distil,reflux,iter,iteres,
      +      xtotdistil,totdistil)
        end if

       if(opmode.eq.2) then
         if(time.ge.switch(order+1)) then
            order=order+1
         end if
         reflux(morvol)=varref(order)
         call dynamic(errnorm,morvol,distil,reflux,iter,iteres,
      +      xtotdistil,totdistil)
       end if
c        ------------------------------------------------------------
c        step 9

c        check if desired amount of distillate with given purity
c        has been withdrawn
c        asign old xpre and ypre values
3006     do 2103 cmp=1,nmc
           xrefpre(cmp)=xref(cmp)
           xfpre(cmp)=xfeed(cmp)
           do 2102 ii=1,numg
             xpre(cmp,ii)=xmolfrac(cmp,ii)
             ypre(cmp,ii)=ymolfrac(cmp,ii)
             xintpre(cmp,ii)=xint(cmp,ii)
             yintpre(cmp,ii)=yint(cmp,ii)
 2102      continue
           xpre(cmp,numg)=xref(cmp)
           ypre(cmp,1)=yboil(cmp)
 2103    continue
         do 2106 ii=1,numg
           blpre(ii)=liqhold(ii)
           bgpre(ii)=gashold(ii)
           mixhlpre(ii)=mixhl(ii)
           mixhvpre(ii)=mixhv(ii)
 2106    continue
         preconhold=conhold
         conhlpre=mixhl(numg)
         rebhl=rebhlnew
         rebhv=rebhvnew
         if(morvol.lt.nmc) then
           xtotdistil=xtotdistil+xref(morvol)*distil*deltatime

c        average more volatile composition in the product collector
           xdisaverage=xtotdistil/totdistil
         else
           xdisaverage=xfeed(nmc)
         end if
```

```
          time=time+deltatime
         if(mod(iter,yazpro).eq.0) then
           call out(prog,time,iter,iteres,morvol,totdistil,
     +         xdisaverage,distil,reflux,qreb,qc,norm,totlhold,totghold)
         end if
          call presdropcal(totpresdrop)


c         writing up the results into modelres file
         if(mod(iter,yazpro).eq.0) then
          if(nmc.eq.3) then
          write(30,9990)time*60,tstill-273.15d0,ttk(int(0.425/deltaz))-
     +         273.15d0,ttk(int(0.85/deltaz))-273.15d0,
     +         ttk(int(1.275/deltaz))-273.15d0,ttk(int(1.7/deltaz))-
     +         273.15d0,totpresdrop,p(1),p(numg),rebhold(1)*1.0d3,
     +         qreb,qc,vboil,xfeed(1),xfeed(2),
     +         ymolfrac(1,numg),ymolfrac(2,numg),
     +         xref(1),xref(2),reflux(morvol)
          write(*,9990)time*60,tstill-273.15d0,ttk(int(0.425/deltaz))-
     +         273.15d0,ttk(int(0.85/deltaz))-273.15d0,
     +         ttk(int(1.275/deltaz))-273.15d0,ttk(int(1.7/deltaz))-
     +         273.15d0,totpresdrop,p(1),p(numg),rebhold(1)*1.0d3,
     +         qreb,qc,vboil,xfeed(1),xfeed(2),
     +         ymolfrac(1,numg),ymolfrac(2,numg),
     +         xref(1),xref(2),reflux(morvol)
          else
          write(30,9991)time*60,tstill-273.15d0,ttk(int(0.425/deltaz))-
     +         273.15d0,ttk(int(0.85/deltaz))-273.15d0,
     +         ttk(int(1.275/deltaz))-273.15d0,ttk(int(1.7/deltaz))-
     +         273.15d0,totpresdrop,p(1),p(numg),rebhold(1)*1.0d3,
     +         qreb,qc,vboil,xfeed(1),
     +         ymolfrac(1,numg),
     +         xref(1),reflux(morvol)
          write(*,9991)time*60,tstill-273.15d0,ttk(int(0.425/deltaz))-
     +         273.15d0,ttk(int(0.85/deltaz))-273.15d0,
     +         ttk(int(1.275/deltaz))-273.15d0,ttk(int(1.7/deltaz))-
     +         273.15d0,totpresdrop,p(1),p(numg),rebhold(1)*1.0d3,
     +         qreb,qc,vboil,xfeed(1),
     +         ymolfrac(1,numg),
     +         xref(1),reflux(morvol)
          end if
          end if
 9990     format(f10.4,1x,5(f7.2,1x),3(f10.5,1x),f10.4,1x,f10.2,1x,f10.2,
     +         f10.5,2(1x,f10.8),2(1x,f10.8),2(1x,f10.8),(1x,f7.4))
 9991     format(f10.4,1x,5(f7.2,1x),3(f10.5,1x),f10.4,1x,f10.2,1x,f10.2,
     +         f10.5,(1x,f10.8),(1x,f10.8),(1x,f10.8),(1x,f7.4))
          iter=iter+1
          artis=0
          iteres=1
          relaxfac=relaxgiven
          if(opmode.eq.3.and.reflux(morvol).ge.99.0d0) then
            goto 2178
          end if
          if(opmode.eq.3.and.reflux(morvol).lt.99.0d0) then
          open(22,file='savedat')
c         write  reflux calculation results  to 'savedat' file
          write(22,7111)tstill,rebhold(1),conhold,qreb,qc,vboil,
     +         totdistil,distil
          write(22,7117)(xfeed(jj),jj=1,nmc)
          write(22,7117)(yboil(jj),jj=1,nmc)
          write(22,6789)((xmolfrac(i,j),j=1,numg),i=1,nmc)
          write(22,6789)((ymolfrac(i,j),j=1,numg),i=1,nmc)
          write(22,6789)((xint(i,j),j=1,numg),i=1,nmc)
          write(22,6789)((yint(i,j),j=1,numg),i=1,nmc)
          write(22,6789)(ttk(j),j=1,numg)
          write(22,6789)(liqrate(j),j=1,numg)
          write(22,6789)(gasrate(j),j=1,numg)
          close (unit=22)
            stepref=4.0d0
            err1=1.0d0
            err2=1.0d0
            goto 2000
          end if
          if(stopcr.eq.2.and.morvol.lt.nmc.and.
     +       xpurity(morvol).lt.xdisaverage) then
            goto 2000

          end if
          if(stopcr.eq.2.and.morvol.eq.nmc.and.
     +       xpurity(morvol).gt.xdisaverage) then
            goto 2000
          end if
          if(stopcr.eq.1.and.elapsed(morvol).ge.(time*60.0d0)) then
            goto 2000
          end if
 2178     print *,'the end of product=',morvol,namecomp(morvol)
          print *,'distillate=',totdistil

c         change the product collector
c         next component is coming
          charge=charge-totdistil
          totdistil=0.0d0
  178   continue
c         ************************************************************

  2001 format (1x,e7.2,4x,e7.3,4x,e10.3,4x,e8.3)
  2002 format ('time (hr)=',e7.2,'    treb(c)=',e7.3,
     +     '    hi(kj/m2 k hr)=',e10.3,'   u(kj/m2 k hr)=',e10.3)

  4000 close (unit=14)
       close (unit=15)
       close (unit=16)
       close (unit=17)
       close (unit=18)
       close (unit=19)
       close (unit=20)
       close (unit=30)
       end


       subroutine dynamic(errnorm,morvol,distil,reflux,iter,iteres,
     +       xtotdistil,totdistil)
       implicit double precision (a-z)
       external fcn,dres1,dres2,dres3,dres4,dres5
       integer numg,lrw,liw,idid,ipar,info(15),morvol,prog,artis
       integer nmc,ii,jj,comp,compon,n,typack,yaztot,yazpro,neq,simstop
       integer j,kk,l,i,iter,cmp,iteres,option,solstart,decide,model
       integer simpro,opmode,objfonk,var,kindex,pacmat,iwork(250)

       include 'param.f'
       include 'comblok.f'

       common/previous/totlhold,totghold,time,temppre,xmolpre,
     +   xintpre,ymolpre,yintpre,prxref,
     +   xrefpre,xfpre,xfprime
       common/dec3/qbinit,qbartis,qbmax,qc,relaxgiven

       dimension temppre(1:numg),xintpre(1:nmc,1:numg)
       dimension prxref(1:nmc),sol(2*numg),yintpre(1:nmc,1:numg)
       dimension v(1:nmc),yprime(2*numg)
       dimension solen(2*numg)
       dimension xfprime(1:nmc),xmolpre(1:nmc,1:numg),xrefpre(1:nmc)
       dimension rwork(42000),ymolpre(1:nmc,1:numg),xfpre(1:nmc)
       dimension xpurity(1:nmc),reflux(1:nmc)
       dimension solprime(2*numg),switch(1:20),varref(1:20)
       relaxfac=relaxgiven
       iteres=1
        if(opmode.eq.3) then
         open(22,file='savedat')
c        read total reflux calculation results from totalref file
         read(22,7111)tstill,rebhold(1),conhold,qreb,qc,vboil,
     +        totdistil,distil
         read(22,7117)(xfeed(jj),jj=1,nmc)
         read(22,7117)(yboil(jj),jj=1,nmc)
         read(22,6789)((xmolfrac(i,j),j=1,numg),i=1,nmc)
         read(22,6789)((ymolfrac(i,j),j=1,numg),i=1,nmc)
```

```
      read(22,6789)((xint(i,j),j=1,numg),i=1,nmc)
      read(22,6789)((yint(i,j),j=1,numg),i=1,nmc)
      read(22,6789)(ttk(j),j=1,numg)
      read(22,6789)(liqrate(j),j=1,numg)
      read(22,6789)(gasrate(j),j=1,numg)
      close (unit=22)
      call propcal
    end if
7111 format(8(d22.16,1x))
7117 format(3(d22.16,1x))
6789 format(30(d22.16,1x))
     artis=0
     liqref=(gasrate(numg)*crosarea-(conhold-preconhold)/
    +       deltatime)/(1.0d0+1.0d0/reflux(morvol))
     distil=liqref/reflux(morvol)
     call propcal
     do 301 ii=1,numg
       liqrate(ii)=liqrate(ii)-distil/crosarea
301  continue

     call  enthal(xfeed,yboil,tstill,presbot,rebhl,rebhv,v)
c      replace liquid rates by liquid rates - distillate rate
c      calculate new xmolfrac,xint,ymolfrac,yint values
2000   do 67 cmp=1,nmc
         do 69 ii=1,numg
           xintpre(cmp,ii)=xint(cmp,ii)
           yintpre(cmp,ii)=yint(cmp,ii)
           xmolpre(cmp,ii)=xmolfrac(cmp,ii)
           ymolpre(cmp,ii)=ymolfrac(cmp,ii)
69       continue
67     continue
       rtol=0.0d0
       atol=1.0d-6
       lrw=42000
       liw=250
       neq=2*numg
       do 117 i=1,2*numgc
         yprime(i)=0.0d0
         sol(i)=0.0d0
117    continue
       do 82 comp=1,nmc
         t=time
         tout=t+deltatime
         do 115 i=1,15
           info(i)=0
115      continue
         info(4)=1
         rwork(1)=tout
c       info(11)=1
         idid=0
c      dassl by petzold application on mass transfer equations
c      initialisation of mass balance equation vector
         kindex=1
         do 116 i=1,numg
           sol(kindex)=xpre(comp,i)
           sol(kindex+1)=ypre(comp,i)
           kindex=kindex+2
116      continue
         sol(2)=yboil(comp)
         sol(2*numg-1)=xref(comp)
         ipar=1


         if(option.eq.3) then
           call masstrforward(model,sol)
           goto 223
         end if
         if(simpro.eq.2) then
            call turev(sol,yprime)
134         call ddassl(dres1,neq,t,sol,yprime,tout,info,rtol,
     +        atol,idid,rwork,lrw,iwork,liw,rpar,ipar,djac1)
           if(idid.eq.1) then
              call ddassl(dres1,neq,t,sol,yprime,tout,info,rtol,
     +             atol,idid,rwork,lrw,iwork,liw,rpar,ipar,djac1)
           end if
         if(idid.eq.(-1).or.idid.eq.(-2)) then
           info(1)=1
           call ddassl(dres1,neq,t,sol,yprime,tout,info,rtol,
     +          atol,idid,rwork,lrw,iwork,liw,rpar,ipar,djac1)
         end if
         if(idid.eq.(-3)) then
           atol=1.0d-6
           info(1)=1
           call ddassl(dres1,neq,t,sol,yprime,tout,info,rtol,
     +          atol,idid,rwork,lrw,iwork,liw,rpar,ipar,djac1)
         end if
         end if
         if (idid.lt.0.or.simpro.eq.1) then
           call  massbal(model,sol)
         end if

c  ****************************************************************

c  asign new xmolfrac, xinterfacial, ymolfrac and yinterfacial
c  mol fractions by matrix solution
 223     l=1
         do 766 j=1,numg
           xmolfrac(comp,j)=sol(l)
           ymolfrac(comp,j)=sol(l+1)

           if(j.eq.1) then
             xint(comp,j)=xmolfrac(comp,j)
             yint(comp,j)=ymolfrac(comp,j)
             goto 678
           end if

           if(j.eq.numg) then
             xint(comp,j)=xmolfrac(comp,j)
             yint(comp,j)=ymolfrac(comp,j)
             goto 678
           end if

           if(comp.eq.nmc) then
             liqtop=0.0d0
             do 878 i=1,nmc-1
               liqtop=liqtop+liqmasstr(i,j)*(xmolfrac(i,j)-xint(i,j))
878          continue
             xint(comp,j)=xmolfrac(comp,j)+liqtop/liqmasstr(comp,j)
           else
             xint(comp,j)=(xmolfrac(comp,j)+gasmasstr(comp,j)/
     +            liqmasstr(comp,j)*ymolfrac(comp,j))/
     +            (gasmasstr(comp,j)/liqmasstr(comp,j)*
     +            keqcons(comp,j)+1.0d0)
           end if
           yint(comp,j)=keqcons(comp,j)*xint(comp,j)

678        l=l+2
766      continue
82     continue

c   ****************************************************************
       call unif(numg,nmc,xmolfrac,xint,ymolfrac,yint)
c   ****************************************************************
c                            step 3

c      calculation of new temperatures in packed section
       do 101 ii=1,numg
         temppre(ii)=ttk(ii)
101    continue
       call newtempcal(xfeed,xref,xint,p,ttk)

c   ****************************************************************
c                            step 4
c      call propcal
```

```
731
732           totlhold=0.0d0
733           totghold=0.0d0
734           do 461 j=1,numg
735             lhold(j)=liqhold(j)*crosarea*deltaz
736             ghold(j)=gashold(j)*crosarea*deltaz
737             totlhold=totlhold+liqhold(j)
738             totghold=totghold+gashold(j)
739    461    continue
740
741           totlhold=(totlhold/numg)*colvol
742           totghold=(totghold/numg)*colvol
743
744
745    c      gas and liquid rates [kg/m2 hr]
746
747    c      new reflux drum holdup [kmol]
748           conhold=volconhold*volrdrum*mlden(numg)/lmixwme(numg)
749
750    c      condenser-reflux drum calculations
751    c      liquid reflux [kmol/hr]
752           liqref=(gasrate(numg)*crosarea-(conhold-preconhold)/
753         +       deltatime)/(1.0d0+1.0d0/reflux(morvol))
754           distil=liqref/reflux(morvol)
755           totdistil=totdistil+distil*(deltatime)
756    c      reflux composition xref(comp)
757           if(conhold.ne.0.0d0) then
758             t=time
759             tout=t+deltatime
760             do 175 i=1,15
761               info(i)=0
762    175      continue
763             info(4)=1
764             info(11)=1
765             rwork(1)=tout
766             do 888 cmp=1,nmc
767               xref(cmp)=xrefpre(cmp)
768    888      continue
769             idid=0
770             call ddassl(dres4,nmc,t,xref,prxref,tout,info,rtol,
771         +         atol,idid,rwork,lrw,iwork,liw,rpar,ipar,djac4)
772           end if
773           topxref=0.0d0
774           do 671 i=1,nmc
775              topxref=xref(i)+topxref
776    671    continue
777           do 688 i=1,nmc
778               xref(i)=xref(i)/topxref
779    688    continue
780           if(conhold.eq.0.0d0) then
781             do 199 comp=1,nmc
782               xref(comp)=ymolfrac(comp,numg)
783    199      continue
784           end if
785           do 919 comp=1,nmc
786             xmolfrac(comp,numg)=xref(comp)
787    919    continue
788
789    c  ************************************************************
790    c                              step 5
791    c     calculate a new set of liquid and vapour molar flow rates
792           do 789 i=1,numg
793             lrpre(i)=liqrate(i)
794             grpre(i)=gasrate(i)
795    789    continue
796             do 342 i=1,numg
797               solen(2*i-1)=liqrate(i)
798               solen(2*i)=gasrate(i)
799               solprime(2*i-1)=0.0d0
800               solprime(2*i)=0.0d0
801    342      continue
802
803           if(option.eq.3) then
804             call forbakenerstart(model,solen)
805             goto 790
806           end if
807           if(simpro.eq.2) then
808             t=time
809             tout=t+deltatime
810             do 195 i=1,15
811               info(i)=0
812    195      continue
813             info(10)=1
814             info(4)=1
815             rwork(1)=tout
816             idid=0
817             call ddassl(dres5,2*numg,t,solen,solprime,tout,info,
818         +         rtol,atol,idid,rwork,lrw,iwork,liw,rpar,ipar,djac5)
819          end if
820
821          if(idid.lt.0.or.simpro.eq.1) then
822             do 372 i=1,numg
823               solen(2*i-1)=liqrate(i)
824               solen(2*i)=gasrate(i)
825    372      continue
826             call enerbal(model,solen)
827          end if
828
829    790     kk=1
830            do 107 ii=1,numg
831              liqrate(ii)=solen(kk)
832              gasrate(ii)=solen(kk+1)
833              kk=kk+2
834    107     continue
835    c       call propcal
836            call  presdropcal(totpresdrop)
837
838    c       ************************************************************
839    c                                  step 6
840    c       calculate the reboiler molar holdup
841            rebhold(1)=charge-totlhold-totghold-conhold-totdistil
842    c       new reboiler composition
843            t=time
844            tout=t+deltatime
845            do 135 i=1,15
846              info(i)=0
847    135     continue
848            info(4)=1
849            info(11)=1
850            rwork(1)=tout
851            do 777 cmp=1,nmc
852              xfeed(cmp)=xfpre(cmp)
853    777     continue
854            idid=0
855            call ddassl(dres3,nmc,t,xfeed,xfprime,tout,info,rtol,
856         +       atol,idid,rwork,lrw,iwork,liw,rpar,ipar,djac3)
857            top=0.0d0
858            do 146 i=1,nmc
859              top=top+xfeed(i)
860    146     continue
861            do 147 i=1,nmc
862              xfeed(i)=xfeed(i)/top
863    147     continue
864
865            call rebtemp(presbot,xfeed,tstill,yboil)
866            do 176 comp=1,nmc
867              ymolfrac(comp,1)=yboil(comp)
868    176     continue
869
870    c ******************************************************************
871    c                                step 8
872    c  calculate condenser and reboiler heat transfer rates
873     ^Icall  enthal(xfeed,yboil,tstill,presbot,
874          +       rebhlnew,rebhvnew,v)
875    c  decide=1 then constant boilup rate is given vboil [kmol/hr]
876    c  decide=2 then qreb [kj/hr] is given and vboil will be calculated.
```

```
c  decide=3 then qreb and vboil both will be calculated.
      if(decide.eq.1) then
c       heat input to the reboiler
        qreb=vboil*(rebhvnew-rebhlnew)-liqrate(1)*crosarea*
     +        (mixhl(1)-rebhlnew)
      end if

      if(decide.eq.2) then
        vboil=(qreb+liqrate(1)*crosarea*(mixhl(1)-rebhlnew))/
     +        (rebhvnew-rebhlnew)
      end if

      if(decide.eq.3) then
c       heat input to the reboiler and vboil will be calculated
        qreb=qbinit+qbartis*time
        if(qreb.gt.qbmax) then
             qreb=qbmax
        end if
        vboil=(qreb+liqrate(1)*crosarea*(mixhl(1)-rebhlnew))/
     +        (rebhvnew-rebhlnew)
      end if


c       condenser heat output qc[kj/hr]
        qc=gasrate(numg)*crosarea*(mixhv(numg)-mixhl(numg))
c     +       -conhold*(mixhl(numg)-conhlpre)/deltatime

c  ****************************************************************
c                              step 7
c  repeat steps 2 through 6 with a corrector step
c  for the same time increment
        normpre=norm
        call normcal(temppre,ttk,norm)
        call matnormcal(xmolpre,xmolfrac,normx)
        call matnormcal(xintpre,xint,normxint)
        call matnormcal(ymolpre,ymolfrac,normy)
        call matnormcal(yintpre,yint,normyint)
        call normcal(lrpre,liqrate,normlr)
        call normcal(grpre,gasrate,normgr)
        write(*,2288)iter,iteres,norm,normlr,normgr,normx,normxint,
     +      normy,normyint
        artis=artis+1

        if(artis.ge.10) then
             artis=0
             relaxfac=relaxfac/1.5d0
        end if
        write(18,2288)iter,iteres,norm,normlr,normgr,normx,normxint,
     +     normy,normyint
 2288 format(i4,1x,i4,7(1x,f15.10))
        if(dabs(norm).ge.errnorm.and.iteres.lt.100) then
          totdistil=totdistil-distil*(deltatime)
          iteres=iteres+1
c       relaxation step
          do 203 j=1,numg
            do 24 comp=1,nmc
              xmolfrac(comp,j)=(1.0d0-relaxfac)*xmolpre(comp,j)+
     +             relaxfac*xmolfrac(comp,j)
              xint(comp,j)=(1.0d0-relaxfac)*xintpre(comp,j)+
     +             relaxfac*xint(comp,j)
              ymolfrac(comp,j)=(1.0d0-relaxfac)*ymolpre(comp,j)+
     +             relaxfac*ymolfrac(comp,j)
              yint(comp,j)=(1.0d0-relaxfac)*yintpre(comp,j)+
     +             relaxfac*yint(comp,j)
 24         continue
            ttk(j)=(1.0d0-relaxfac)*temppre(j)+relaxfac*ttk(j)
            liqrate(j)=(1.0d0-relaxfac)*lrpre(j)+relaxfac*liqrate(j)
            gasrate(j)=(1.0d0-relaxfac)*grpre(j)+relaxfac*gasrate(j)
 203      continue
          goto 2000
        endif
        artis=0
        return
        end

      subroutine massbal(model,sol)
c**********************************************************************
c       solving dynamic mass balance equations
c       time derivative by finite difference method [backward dif.],
c       axial spatial [forward and backward dif.]
c**********************************************************************


      implicit double precision (a-z)
      integer numg,model,nmc,comp,j,i
      include 'param.f'

      common/disperblok/laxdis(1:numg),gaxdis(1:numg)
      common/holdupblok/volrdrum,volconhold,preconhold,
     +     conhold,rebhold(1),liqhold(1:numg),gashold(1:numg)
      common/flowrate/liqrate(1:numg),gasrate(1:numg)
      common/mastran/liqmasstr(1:nmc,1:numg),gasmasstr(1:nmc,1:numg)
      common/propla/intarea(1:numg),keqcons(1:nmc,1:numg)
      common/start1/deltatime,deltaz,comp
      common/boundary/yboil(1:nmc),xref(1:nmc)
      common/premolfrac/xpre(1:nmc,1:numg),ypre(1:nmc,1:numg)
      common/parastudy/alfaky,alfakx,beta1,beta2

      dimension zmat(2*numg,2*numg),gval(2*numg),sol(2*numg)

      do 11 i=1,2*numg
        do 22 j=1,2*numg
          zmat(i,j)=0.0d0
 22     continue
        gval(i)=0.0d0
        sol(i)=0.0d0
 11   continue

c       reboiler-column interface

      zmat(1,1)=-3.0d0
      zmat(1,3)=4.0d0
      zmat(1,5)=-1.0d0

      zmat(2,2)=1.0d0

      gval(2)=yboil(comp)

c       column-condenser interface
      zmat(2*numg-1,2*numg-1)=1.0d0

      gval(2*numg-1)=xref(comp)


      zmat(2*numg,2*numg-4)=1.0d0
      zmat(2*numg,2*numg-2)=-4.0d0
      zmat(2*numg,2*numg)=3.0d0


c       column section
      do 93 j=2,numg-1

        coff=gasmasstr(comp,j)*intarea(j)/
     +       (gasmasstr(comp,j)/liqmasstr(comp,j)*keqcons(comp,j)+1.0d0)

        zmat(2*j-1,2*(j)-1)=liqhold(j)/deltatime+liqrate(j)/deltaz
     +       +coff*keqcons(comp,j)

        if(j.ne.numg-1) then
          zmat(2*j-1,2*(j+1)-1)=-liqrate(j)/(deltaz)
        end if

        zmat(2*j-1,2*j)=-coff

        gval(2*j-1)=liqhold(j)/deltatime*xpre(comp,j)
```

```
1023          if(j.eq.numg-1) then
1024            gval(2*j-1)=gval(2*j-1)+xref(comp)*(liqrate(j)/(deltaz))
1025          end if
1026
1027          gval(2*j)=gashold(j)/deltatime*ypre(comp,j)
1028          if(j.ne.2) then
1029            zmat(2*j,2*(j-1))=-gasrate(j)/(deltaz)
1030          end if
1031
1032          if(j.eq.2) then
1033            gval(2*j)=gval(2*j)+yboil(comp)*(gasrate(j)/(deltaz))
1034          end if
1035
1036          zmat(2*j,2*j-1)=-coff*keqcons(comp,j)
1037
1038
1039          zmat(2*j,2*j)=coff+gasrate(j)/deltaz+gashold(j)/deltatime
1040
1041
1042          if(model.eq.2) then
1043            zmat(2*j-1,2*(j-1)-1)=zmat(2*j-1,2*(j-1)-1)
1044       +         -0.5d0*laxdis(j)*liqhold(j)/(deltaz**2.0d0)
1045
1046            zmat(2*j-1,2*(j)-1)=zmat(2*j-1,2*(j)-1)+
1047       +         laxdis(j)*liqhold(j)/(deltaz**2.0d0)
1048            if(j.ne.numg-1) then
1049              zmat(2*j-1,2*(j+1)-1)=zmat(2*j-1,2*(j+1)-1)
1050       +           -laxdis(j)*liqhold(j)/(deltaz**2.0d0)
1051            end if
1052            gval(2*j-1)=gval(2*j-1)
1053       +         +0.5d0*laxdis(j)*liqhold(j)/(deltaz**2.0d0)*
1054       +         (xpre(comp,j+1)-2.0d0*xpre(comp,j)+xpre(comp,j-1))
1055
1056            if(j.eq.numg-1) then
1057              gval(2*j-1)=gval(2*j-1)
1058       +           +(laxdis(j)*liqhold(j)/(deltaz**2.0d0))*xref(comp)
1059            end if
1060
1061            gval(2*j)=gval(2*j)
1062       +         +0.5d0*gaxdis(j)*gashold(j)/(deltaz**2.0d0)*
1063       +         (ypre(comp,j+1)-2.0d0*ypre(comp,j)+ypre(comp,j-1))
1064            if(j.ne.2) then
1065              zmat(2*j,2*(j-1))=zmat(2*j,2*(j-1))
1066       +           -0.5d0*gaxdis(j)*gashold(j)/(deltaz**2.0d0)
1067            end if
1068
1069            if(j.eq.2) then
1070              gval(2*j)=gval(2*j)
1071       +           +(gaxdis(j)*gashold(j)/(deltaz**2.0d0))*yboil(comp)
1072            end if
1073
1074
1075            zmat(2*j,2*j)=zmat(2*j,2*j)
1076       +         +gaxdis(j)*gashold(j)/(deltaz**2.0d0)
1077
1078            zmat(2*j,2*(j+1))=zmat(2*j,2*(j+1))
1079       +         -0.5d0*gaxdis(j)*gashold(j)/(deltaz**2.0d0)
1080
1081          end if
1082
1083
1084   93   continue
1085
1086        call gauss(2*numg,numg,zmat,gval,sol)
1087
1088   c      kayit=0
1089   c      do 90 i=1,2*numg
1090   c      do 91 j=1,2*numg
1091   c      if(zmat(i,j).ne.0.0d0) then
1092   c      kayit=kayit+1
1093   c      write(2,2345)kayit,i,j,zmat(i,j)
1094   c      write(*,2345)kayit,i,j,zmat(i,j)
1095   c      end if
1096   c91    continue
1097   c90    continue
1098   c      write(2,*)
1099   c      write(3,2346)(gval(i),i=1,2*numg)
1100   c      write(*,2346)(gval(i),i=1,2*numg)
1101   c      write(3,2347)(sol(i),i=1,2*numg)
1102   c2345  format(i5,i5,i5,1x,(d22.15))
1103   c2346  format('gval=',30d22.15)
1104   c2347  format('solu=',30d22.15)
1105          return
1106          end
1107
1108
1109          subroutine enerbal(model,solen)
1110   c**********************************************************************
1111   c     it solves dynamic energy balance equations for new liquid and
1112   c     vapour molar flux values [kmol/m2 h]
1113   c     time derivative is discritized by backward finite difference
1114   c     spatial derivative terms are discritized by forward finite
1115   c     difference for liquid and backward finite difference for vapour side
1116   c**********************************************************************
1117
1118          implicit double precision (a-z)
1119          integer numg,nmc,j,ll,model,comp
1120          include 'param.f'
1121          dimension zmaten(2*numg,2*numg),gvalen(2*numg)
1122          dimension solen(2*numg)
1123          common/holdupblok/vl,vc,pr,conhold,rebhold(1),
1124         +     liqhold(1:numg),gashold(1:numg)
1125          common/disperblok/laxdis(1:numg),gaxdis(1:numg)
1126          common/prop1a/intarea(1:numg),keqcons(1:nmc,1:numg)
1127          common/start1/deltatime,deltaz,comp
1128          common/enthalblk/mixhl(1:numg),mixhv(1:numg),
1129         +     mixhlpre(1:numg),mixhvpre(1:numg)
1130          common/prehold/blpre(1:numg),bgpre(1:numg)
1131          common/start3/colvol,charge,specarea,voidfrac,pacsurten,dp,vboil,
1132         +     liqref,relaxfac,eqdia,crosarea
1133
1134   c       calculation of new total liquid and vapour flow rates
1135   c       throughout the column
1136
1137          do 2 j=1,2*numg
1138            do 3 ll=1,2*numg
1139              zmaten(j,ll)=0.0d0
1140    3       continue
1141            gvalen(j)=0.0d0
1142            solen(j)=0.0d0
1143    2     continue
1144
1145   c       ------------------------------------------------------
1146   c       reboiler - column interface
1147          zmaten(1,1)=-3.0d0
1148          zmaten(1,3)=4.0d0
1149          zmaten(1,5)=-1.0d0
1150
1151          zmaten(2,2)=1.0d0
1152          gvalen(2)=vboil/crosarea
1153
1154   c       column-condenser interface
1155          zmaten(2*numg-1,2*numg-1)=1.0d0
1156          gvalen(2*numg-1)=liqref/crosarea
1157
1158          zmaten(2*numg,2*numg-4)=1.0d0
1159          zmaten(2*numg,2*numg-2)=-4.0d0
1160          zmaten(2*numg,2*numg)=3.0d0
1161
1162
1163   c       column
1164          dlt=deltatime
1165
1166          do 104 j=2,numg-1
1167            gvalen(2*j-1)=-gashold(j)*(mixhv(j)-mixhvpre(j))
1168         +       -mixhv(j)*(gashold(j)-bgpre(j))
```

```
     +       -liqhold(j)*(mixhl(j)-mixhlpre(j))
     +       -mixhl(j)*(liqhold(j)-blpre(j))
          if(j.ne.2) then
            zmaten(2*j-1,2*(j-1))=-dlt*mixhv(j)/deltaz
          else
            gvalen(2*j-1)=gvalen(2*j-1)+
     +             dlt*mixhv(j)/deltaz*vboil/crosarea
          end if

          zmaten(2*j-1,2*j-1)=-dlt*(mixhl(j+1)-mixhl(j))/deltaz+
     +       dlt*mixhl(j)/deltaz
          zmaten(2*j-1,2*j)=dlt*(mixhv(j)-mixhv(j-1))/deltaz+
     +       dlt*mixhv(j)/deltaz

          if(j.ne.numg-1) then
            zmaten(2*j-1,2*(j+1)-1)=-dlt*mixhl(j)/deltaz
          else
              gvalen(2*j-1)=gvalen(2*j-1)+
     +          dlt*mixhl(j)/deltaz*liqref/crosarea
            end if

           gvalen(2*j)=(-liqhold(j)+blpre(j)-gashold(j)+
     +         bgpre(j))

          if(j.ne.2) then
            zmaten(2*j,2*(j-1))=-dlt*1.0d0/deltaz
          else
            gvalen(2*j)=gvalen(2*j)+dlt*1.0d0/deltaz*vboil/crosarea
          end if

          zmaten(2*j,2*j-1)=dlt*1.0d0/deltaz
          zmaten(2*j,2*j)=dlt*1.0d0/deltaz
          if(j.ne.numg-1) then
            zmaten(2*j,2*(j+1)-1)=-dlt*1.0d0/deltaz
          else
            gvalen(2*j)=gvalen(2*j)+dlt*1.0d0/deltaz*liqref/crosarea
          end if

          if(model.eq.2) then
           gvalen(2*j-1)=gvalen(2*j-1)
     +          +dlt*(laxdis(j)*mixhl(j)*(0.5d0/(deltaz**2.0d0))*
     +          (liqhold(j+1)-2.0d0*liqhold(j)+liqhold(j-1)+
     +          blpre(j+1)-2.0d0*blpre(j)+blpre(j-1))
     +          +gaxdis(j)*mixhv(j)*(0.5d0/(deltaz**2.0d0))*
     +          (gashold(j+1)-2.0d0*gashold(j)+gashold(j-1)+
     +          bgpre(j+1)-2.0d0*bgpre(j)+bgpre(j-1)))

           gvalen(2*j)=gvalen(2*j)
     +         +dlt*(laxdis(j)*(0.5d0/(deltaz**2.0d0))*
     +          (liqhold(j+1)-2.0d0*liqhold(j)+liqhold(j-1)
     +         +blpre(j+1)-2.0d0*blpre(j)+blpre(j-1))
     +         +gaxdis(j)*(0.5d0/(deltaz**2.0d0))*
     +         (gashold(j+1)-2.0d0*gashold(j)+gashold(j-1)
     +         +bgpre(j+1)-2.0d0*bgpre(j)+bgpre(j-1)))

        end if
 104  continue


c        do 90 i=1,2*numg
c        do 91 j=1,2*numg
c        if(zmaten(i,j).ne.0.0d0) then
c        write(*,2345)i,j,zmaten(i,j)
c        end if
c91      continue
c90      continue
c        write(2,*)
c        write(*,2346)(gvalen(i),i=1,2*numg)
c2345    format(i5,i5,1x,e20.10)
c2346    format('gval=',2e20.10)

c       calculation of new liquid and vapour flow rates
        call gauss(2*numg,numg,zmaten,gvalen,solen)
        return
        end

        subroutine enerstart(model,solen)
c*********************************************************************
c       it solves steady state energy balance equations for new
c       liquid and vapour molar flux values [kmol/m2/h]
c       spatial derivative terms are discritized by central finite
c       difference
c*********************************************************************

        implicit double precision (a-z)
        integer numg,nmc,ll,j,model
        include 'param.f'
        dimension zmaten(2*numg,2*numg),gvalen(2*numg)
        dimension solen(2*numg)

        common/holdupblok/vl,vc,pr,conhold,rebhold(1),
     +       liqhold(1:numg),gashold(1:numg)
        common/disperblok/laxdis(1:numg),gaxdis(1:numg)
        common/flowrate/liqrate(1:numg),gasrate(1:numg)
        common/propla/intarea(1:numg),keqcons(1:nmc,1:numg)
        common/start1/deltatime,deltaz,comp
        common/enthalblk/mixhl(1:numg),mixhv(1:numg),
     +       mixhlpre(1:numg),mixhvpre(1:numg)
        common/prehold/blpre(1:numg),bgpre(1:numg)
        common/start3/colvol,charge,specarea,voidfrac,pacsurten,
     +       dp,vboil,liqref,relaxfac,eqdia,crosarea

c       calculation of new total liquid and vapour flow rates
c       throughout the column
        do 2 j=1,2*numg
          do 3 ll=1,2*numg
            zmaten(j,ll)=0.0d0
 3        continue
          gvalen(j)=0.0d0
        solen(j)=0.0d0
 2    continue

c       ------------------------------------------------------
c       reboiler - column interface
      zmaten(1,1)=-3.0d0
      zmaten(1,3)=4.0d0
      zmaten(1,5)=-1.0d0

      zmaten(2,2)=1.0d0
      gvalen(2)=vboil/crosarea
c       ------------------------------------------------------
c       column
      do 1 j=2,numg-1
        zmaten(2*j-1,2*j-3)=-(mixhl(j))/(2.0d0*deltaz)
        zmaten(2*j-1,2*j-2)=mixhv(j)/(2.0d0*deltaz)
        zmaten(2*j-1,2*j-1)=(mixhl(j+1)-mixhl(j-1))/(2.0d0*deltaz)
        zmaten(2*j-1,2*j)=-(mixhv(j+1)-mixhv(j-1))/(2.0d0*deltaz)
        zmaten(2*j-1,2*j+1)=mixhl(j)/(2.0d0*deltaz)
        zmaten(2*j-1,2*j+2)=-mixhv(j)/(2.0d0*deltaz)


        zmaten(2*j,2*j-3)=-1.0d0
        zmaten(2*j,2*j-2)=1.0d0
        zmaten(2*j,2*j+1)=1.0d0
        zmaten(2*j,2*j+2)=-1.0d0


        if(j.eq.2) then
          zmaten(2*j-1,2*j-2)=0.0d0
          gvalen(2*j-1)=gvalen(2*j-1)-mixhv(j)/
     +         (2.0d0*deltaz)*vboil/crosarea

          zmaten(2*j,2*j-2)=0.0d0
          gvalen(2*j)=gvalen(2*j)-vboil/crosarea
        end if
```

```
          if(j.eq.numg-1) then
            zmaten(2*j-1,2*j+1)=0.0d0
            gvalen(2*j-1)=gvalen(2*j-1)-mixhl(j)/
     +           (2.0d0*deltaz)*liqref/crosarea

            zmaten(2*j,2*j+1)=0.0d0
            gvalen(2*j)=gvalen(2*j)-liqref/crosarea
          end if
          if(model.eq.2) then
            gvalen(2*j-1)=gvalen(2*j-1)
     +           -(laxdis(j)*mixhl(j)*(liqhold(j+1)-
     +           2.0d0*liqhold(j)+liqhold(j-1))/(2.0d0*deltaz)+
     +           gaxdis(j)*mixhv(j)*(gashold(j+1)-
     +           2.0d0*gashold(j)+gashold(j-1))/(2.0d0*deltaz))
            gvalen(2*j)=gvalen(2*j)
     +           -(laxdis(j)*(liqhold(j+1)-
     +           2.0d0*liqhold(j)+liqhold(j-1))/(2.0d0*deltaz)+
     +           gaxdis(j)*(gashold(j+1)-
     +           2.0d0*gashold(j)+gashold(j-1))/(2.0d0*deltaz))
          end if
 1      continue
c         ------------------------------------------------------
c         column - condenser interface
        zmaten(2*numg-1,2*numg-1)=1.0d0
        gvalen(2*numg-1)=liqref/crosarea


        zmaten(2*numg,2*numg-4)=1.0d0
        zmaten(2*numg,2*numg-2)=-4.0d0
        zmaten(2*numg,2*numg)=3.0d0
c         ------------------------------------------------------


        call gauss(2*numg,numg,zmaten,gvalen,solen)
c         calculation of new liquid and vapour flow rates
        return
        end



        subroutine forbakenerstart(model,solen)
c*********************************************************************
c         it solves steady state energy balance equations for new
c         liquid and vapour molar flux values [kmol/m2/h]
c         spatial derivative terms are discritized by forward and
c         backward finite difference
c*********************************************************************

        implicit double precision (a-z)
        integer numg,nmc,ll,j,model
        include 'param.f'
        dimension zmaten(2*numg,2*numg),gvalen(2*numg)
        dimension solen(2*numg)

        common/holdupblok/vl,vc,pr,conhold,rebhold(1),
     +       liqhold(1:numg),gashold(1:numg)
        common/disperblok/laxdis(1:numg),gaxdis(1:numg)
        common/flowrate/liqrate(1:numg),gasrate(1:numg)
        common/prop1a/intarea(1:numg),keqcons(1:nmc,1:numg)
        common/start1/deltatime,deltaz,comp
        common/enthalblk/mixhl(1:numg),mixhv(1:numg),
     +       mixhlpre(1:numg),mixhvpre(1:numg)
        common/prehold/blpre(1:numg),bgpre(1:numg)
        common/start3/colvol,charge,specarea,voidfrac,pacsurten,dp,vboil,
     +       liqref,relaxfac,eqdia,crosarea

c         calculation of new total liquid and vapour flow rates
c         throughout the column
        do 2 j=1,2*numg
          do 3 ll=1,2*numg
            zmaten(j,ll)=0.0d0
 3        continue
          gvalen(j)=0.0d0
          solen(j)=0.0d0
 2      continue

c         ------------------------------------------------------
c         reboiler - column interface
        zmaten(1,1)=-3.0d0
        zmaten(1,3)=4.0d0
        zmaten(1,5)=-1.0d0

        zmaten(2,2)=1.0d0
        gvalen(2)=vboil/crosarea
c         ------------------------------------------------------
c         column
        do 1 j=2,numg-1
          zmaten(2*j-1,2*j-2)=mixhv(j)/(deltaz)
          zmaten(2*j-1,2*j-1)=(mixhl(j+1)-2.0d0*mixhl(j))/(deltaz)
          zmaten(2*j-1,2*j)=-(2.0d0*mixhv(j)-mixhv(j-1))/(deltaz)
          zmaten(2*j-1,2*j+1)=mixhl(j)/(deltaz)

          zmaten(2*j,2*j-2)=1.0d0
          zmaten(2*j,2*j-1)=-1.0d0
          zmaten(2*j,2*j)=-1.0d0
          zmaten(2*j,2*j+1)=1.0d0


          if(j.eq.2) then
            zmaten(2*j-1,2*j-2)=0.0d0
            gvalen(2*j-1)=gvalen(2*j-1)-mixhv(j)/(deltaz)*vboil/crosarea

            zmaten(2*j,2*j-2)=0.0d0
            gvalen(2*j)=gvalen(2*j)-vboil/crosarea
          end if

          if(j.eq.numg-1) then
            zmaten(2*j-1,2*j+1)=0.0d0
            gvalen(2*j-1)=gvalen(2*j-1)-mixhl(j)/(deltaz)*liqref/crosarea

            zmaten(2*j,2*j+1)=0.0d0
            gvalen(2*j)=gvalen(2*j)-liqref/crosarea
          end if
          if(model.eq.2) then
            gvalen(2*j-1)=gvalen(2*j-1)
     +         -(laxdis(j)*mixhl(j)*(liqhold(j+1)-
     +           2.0d0*liqhold(j)+liqhold(j-1))/(2.0d0*deltaz)+
     +           gaxdis(j)*mixhv(j)*(gashold(j+1)-
     +           2.0d0*gashold(j)+gashold(j-1))/(2.0d0*deltaz))
            gvalen(2*j)=gvalen(2*j)
     +           -(laxdis(j)*(liqhold(j+1)-
     +           2.0d0*liqhold(j)+liqhold(j-1))/(2.0d0*deltaz)+
     +           gaxdis(j)*(gashold(j+1)-
     +           2.0d0*gashold(j)+gashold(j-1))/(2.0d0*deltaz))
          end if
 1      continue
c         ------------------------------------------------------
c         column - condenser interface
        zmaten(2*numg-1,2*numg-1)=1.0d0
        gvalen(2*numg-1)=liqref/crosarea


        zmaten(2*numg,2*numg-4)=1.0d0
        zmaten(2*numg,2*numg-2)=-4.0d0
        zmaten(2*numg,2*numg)=3.0d0
c         ------------------------------------------------------


        call gauss(2*numg,numg,zmaten,gvalen,solen)
c         calculation of new liquid and vapour flow rates
        return
        end


        subroutine propcal
c*********************************************************************
```

```
c     it calls physical, thermodynamic and transfer property routines
c     for component specifications, temperature and pressure
c*********************************************************************

      implicit double precision (a-z)
      integer nmc,numg,stage,comp,option,i,j
      integer yaztot,yazpro,compon,decide,model
      include 'param.f'
      include 'comblok.f'

      dimension newliqvis(1:nmc,1:numg),newmlvis(1:numg)
      dimension v(1:nmc)
      dimension xi(1:nmc),y(1:nmc),x(1:nmc),keq(1:nmc)
      dimension ldensity(1:nmc),gden(1:nmc),lvis(1:nmc),gvis(1:nmc)
      dimension lsur(1:nmc),mixgdif(1:nmc),mixldif(1:nmc)
      character *15 namecomp(1:nmc)

c       calculation of physical properties and holdups
      do 23 stage=1,numg
        lmixwme(stage)=0.0d0
        vmixwme(stage)=0.0d0
^Ido 5 comp=1,nmc
          lmixwme(stage)=lmixwme(stage)+xmolfrac(comp,stage)*wme(comp)
          vmixwme(stage)=vmixwme(stage)+ymolfrac(comp,stage)*wme(comp)
          tr(comp,stage)=ttk(stage)/tc(comp)
          tbr(comp,stage)=(tbp(comp)+273.15d0)/tc(comp)
          pr(comp,stage)=p(stage)/pc(comp)
 5      continue
        grate(stage)=gasrate(stage)*vmixwme(stage)
        lrate(stage)=liqrate(stage)*lmixwme(stage)
 23   continue


      do 50 stage=1,numg
        temp=ttk(stage)
        pres=p(stage)
        do 60 comp=1,nmc
          xi(comp)=xint(comp,stage)
          y(comp)=ymolfrac(comp,stage)
          x(comp)=xmolfrac(comp,stage)
 60     continue

^Icall kval(stage,temp,pres,xi,actliq,keq)
^Icall subdensity(stage,temp,pres,ldensity)
^Icall sgasdensity(stage,temp,pres,gden)
^Icall sliqviscos(temp,lvis)
^Icall sgasviscos(temp,gvis)
^Icall sliqsurten(stage,temp,x,lsur,mixlst)
^Icall smgdiffus(temp,pres,y,mixgdif)
        do 70 comp=1,nmc
          keqcons(comp,stage)=keq(comp)
          liqden(comp,stage)=ldensity(comp)
          gasden(comp,stage)=gden(comp)
          liqvis(comp,stage)=lvis(comp)
          gasvis(comp,stage)=gvis(comp)
          liqsur(comp,stage)=lsur(comp)
          mgdif(comp,stage)=mixgdif(comp)
 70     continue
^Icall mixsgasviscos(temp,y,mixgvis)
        mgvis(stage)=mixgvis
        mlst(stage)=mixlst
^Icall  enthal(x,y,temp,pres,mixhliq,mixhvap,v)
        mixhl(stage)=mixhliq
        mixhv(stage)=mixhvap
 50   continue
      call mpromat(numg,nmc,xmolfrac,liqden,mlden)
      call mpromat(numg,nmc,ymolfrac,gasden,mgden)
      do 44 j=1,numg
        do 45 i=1,nmc
          newliqvis(i,j)=dlog10(liqvis(i,j))
 45     continue
 44   continue
      call mpromat(numg,nmc,xmolfrac,newliqvis,newmlvis)
      do 43 i=1,numg
^Imlvis(i)=10.0d0**(newmlvis(i))
 43   continue
      do 34 stage=1,numg
        call smldiffus(stage,temp,pres,x,mixldif)
        do 75 comp=1,nmc
          mldif(comp,stage)=mixldif(comp)
 75     continue
 34   continue
      call specintarea(specarea,lrate,pacsurten,intarea)

c       calculation of mass transfer coefficients [kmol/m2 hr]
      call sgasmastran(specarea,dp,gasmasstr)
      call sliqmastran(specarea,intarea,dp,liqmasstr)
      call axial(crosarea,dp,laxdis,gaxdis)



c       ************************************************
      if(option.eq.1) then
^Icall holdup(lrate,eqdia,voidfrac,liqhold,gashold)
      end if
      if(option.eq.2) then
        do 7 i=1,numg
          liqhold(i)=vollhold*mlden(i)/lmixwme(i)
          gashold(i)=volghold*mgden(i)/vmixwme(i)
 7      continue
      end if
      if(option.eq.3) then
        do 751 i=1,numg
          liqhold(i)=0.0d0
          gashold(i)=0.0d0
 751    continue
      end if
      return
      end


      subroutine initialise
c*********************************************************************
c     it sets zero to program variables at the beginning of the simulation
c*********************************************************************

      implicit double precision (a-z)
      integer i,j,numg,nmc,compon
      integer yaztot,yazpro,option,decide,model
      include 'param.f'
      include 'comblok.f'

c       ***********************************************
c       initiation of the variables
      do 4 i=1,19
^Iconeq(i,4)=0.0d0
 4    continue
      do 22 j=1,numg
        intarea(j)=0.0d0
        mgvis(j)=0.0d0
        mlvis(j)=0.0d0
        mlst(j)=0.0d0
        ghold(j)=0.0d0
        lhold(j)=0.0d0
        mixhl(j)=0.0d0
        mgden(j)=0.0d0
        mlden(j)=0.0d0
        mixhv(j)=0.0d0
        mixhlpre(j)=0.0d0
        mixhvpre(j)=0.0d0
        blpre(j)=0.0d0
        bgpre(j)=0.0d0
        lmixwme(j)=0.0d0
        vmixwme(j)=0.0d0
        laxdis(j)=0.0d0
        gaxdis(j)=0.0d0
```

```
1607          liqhold(j)=0.0d0
1608          gashold(j)=0.0d0
1609          liqrate(j)=0.0d0
1610          gasrate(j)=0.0d0
1611          ttk(j)=0.0d0
1612    22    continue
1613          do 78 i=1,nmc
1614            chtr(i)=0.0d0
1615            yboil(i)=0.0d0
1616            xref(i)=0.0d0
1617            do 34 j=1,numg
1618              zvap(i,j)=0.0d0
1619              xpre(i,j)=0.0d0
1620              ypre(i,j)=0.0d0
1621              keqcons(i,j)=0.0d0
1622              gasden(i,j)=0.0d0
1623              liqden(i,j)=0.0d0
1624              xmolfrac(i,j)=0.0d0
1625              ymolfrac(i,j)=0.0d0
1626              tr(i,j)=0.0d0
1627              tbr(i,j)=0.0d0
1628              pr(i,j)=0.0d0
1629              liqmasstr(i,j)=0.0d0
1630              gasmasstr(i,j)=0.0d0
1631              gasvis(i,j)=0.0d0
1632              liqvis(i,j)=0.0d0
1633              mgdif(i,j)=0.0d0
1634              mldif(i,j)=0.0d0
1635              liqsur(i,j)=0.0d0
1636              xint(i,j)=0.0d0
1637              yint(i,j)=0.0d0
1638    34      continue
1639    78    continue
1640          rebhold(1)=0.0d0
1641          return
1642          end
1643
1644          subroutine start(qreb)
1645    c*************************************************************************
1646    c     total reflux calculations
1647    c     steady state period
1648    c     it produces initial values for the product period
1649    c*************************************************************************
1650
1651          implicit double precision (a-z)
1652          integer numg
1653          integer nmc,ii,jj,comp,compon,yaztot,prog
1654          integer j,kk,l,i,iter,cmp,option,yazpro,decide,model
1655
1656          include 'param.f'
1657          include 'comblok.f'
1658          common/dec3/qbinit,qbartis,qbmax
1659
1660          dimension sol(2*numg)
1661          dimension temppre(1:numg)
1662          dimension yintpre(1:nmc,1:numg)
1663          dimension v(1:nmc)
1664          dimension xb0(1:nmc)
1665          dimension xintpre(1:nmc,1:numg)
1666          dimension solen(2*numg)
1667
1668          character *15 namecomp(1:nmc)
1669          character *20 totalref
1670          character *20 moltotal
1671          character *20 flowtotal
1672          character *20 phytotal
1673          character *20 errortotal
1674          character *20 reptotal
1675
1676    c     open(2,file='matrix.dat')
1677    c     open(3,file='solution.dat')
1678
1679          open(24,file=moltotal)
1680          open(25,file=phytotal)
1681          open(26,file=flowtotal)
1682          open(28,file=errortotal)
1683          open(19,file=totalref)
1684
1685          write(24,11111)
1686          write(25,11112)
1687          write(26,11113)
1688          write(28,11115)
1689    11111 format(/,/,10x,'Total Reflux Steady State Calculation Results',/,
1690         +     5x,'Bulk and Interfacial Mol Fraction Values of Liquid and ',
1691         +     'Vapour Phases')
1692    11112 format(/,/,10x,'Total Reflux Steady State Calculation Results',/,
1693         +     5x,'Physical, Thermodynamic and Transfer Properties of ',
1694         +'Individual Components',/,30x,'and the Mixture')
1695    11113 format(/,/,10x,'Total Reflux Steady State Calculation Results',/,
1696         +     5x,'Molar and Mass Flux Values of Liquid and ',
1697         +     'Vapour Phases')
1698    11115 format(/,/,10x,'Total Reflux Steady State Calculation Results',/,
1699         +     5x,'Error Values at the end of each Iteration')
1700    c     *********************************************************
1701    c     startup procedure - total reflux conditions
1702    c     distillate rate [kmol/hr]
1703    c     distil=0.0d0
1704          prog=1
1705          relaxfac=1.0d0
1706    c     ***********************************************
1707          time=0.0d0
1708    c      write(*,22222)time
1709    22222 format('time=',f15.10)
1710    c     ***********************************************
1711
1712    c     ---------------------------------------------------------
1713    c     step 1
1714          iter=1
1715    c     keep the initial composition of the charge saved
1716          do 11 comp=1,nmc
1717            xb0(comp)=xfeed(comp)
1718     11   continue
1719    c     reboiler temperature and vapour mol fraction calculations
1720          call rebtemp(presbot,xfeed,tstill,yboil)
1721
1722          call  enthal(xfeed,yboil,tstill,presbot,
1723         +     rebhl,rebhv,v)
1724
1725    c       ************************************************************************
1726    c       step 2
1727    c  decide=1 then constant boilup rate is given vboil [kmol/hr]
1728    c  decide=2 then qreb [kj/hr] is given and vboil will be calculated.
1729    c  decide=3 then qreb=qbinit and vboil will be calculated.
1730          if(decide.eq.2) then
1731            vboil=qreb/(rebhv-rebhl)
1732          end if
1733          if(decide.eq.3) then
1734            qreb=qbinit
1735            vboil=qreb/(rebhv-rebhl)
1736          end if
1737    c       initialisation of packed section temperature, gas molar flow rates
1738    c       and vapour mol fractions
1739    c^Igas rate [kmol/m2 hr]
1740          do 674 ii=1,numg
1741            ttk(ii)=tstill
1742      ^Igasrate(ii)=vboil/crosarea
1743            do 28 comp=1,nmc
1744              ymolfrac(comp,ii)=yboil(comp)
1745     28     continue
1746     674  continue
1747
1748    c       *****************************************************
1749    c       step 3
1750    c       condenser-reflux drum calculations
1751    c       liquid reflux [kmol/hr]
1752          liqref=gasrate(numg)*crosarea
```

```
      do 181 comp=1,nmc
        xref(comp)=ymolfrac(comp,numg)
 181  continue
c       assumptions for packed section liquid flow rates and
c       mol fractions

c^Iliquid rate [kmol/m2 hr]
      do 675 ii=1,numg
^Iliqrate(ii)=liqref/crosarea
        do 676 comp=1,nmc
          xmolfrac(comp,ii)=xref(comp)
          xint(comp,ii)=xmolfrac(comp,ii)
          yint(comp,ii)=ymolfrac(comp,ii)
 676    continue
 675  continue

c       normalisation of mol fractions to 1.0
      call unif(numg,nmc,xmolfrac,xint,ymolfrac,yint)

c       correction of temperature assumptions
      call newtempcal(xfeed,xref,xint,p,ttk)
c       ************************************************
c       step 4
c       calculation of physical properties
      call propcal




      totlhold=0.0d0
      totghold=0.0d0
      do 46 ii=1,numg
        lhold(ii)=liqhold(ii)*crosarea*deltaz
        ghold(ii)=gashold(ii)*crosarea*deltaz
        totlhold=totlhold+liqhold(ii)
        totghold=totghold+gashold(ii)
 46   continue
      totlhold=totlhold/numg*colvol
      totghold=totghold/numg*colvol


      conhold=volconhold*volrdrum*mlden(numg)/lmixwme(numg)

c       ************************************************
c       step 5
c       new mol fractions

c       asign old xpre and ypre values
 1000 do 103 cmp=1,nmc
        do 102 ii=1,numg
          xpre(cmp,ii)=xmolfrac(cmp,ii)
          ypre(cmp,ii)=ymolfrac(cmp,ii)
          xintpre(cmp,ii)=xint(cmp,ii)
          yintpre(cmp,ii)=yint(cmp,ii)
 102    continue
 103  continue

      do 82 comp=1,nmc

c        call masstransfer(model,sol)
        call masstrforward(model,sol)

        l=1
        do 766 j=1,numg
          xmolfrac(comp,j)=sol(l)
          ymolfrac(comp,j)=sol(l+1)

          if(j.eq.1) then
            xint(comp,j)=xmolfrac(comp,j)
            yint(comp,j)=ymolfrac(comp,j)
             goto 678
           end if


          if(j.eq.numg) then
            xint(comp,j)=xmolfrac(comp,j)
            yint(comp,j)=ymolfrac(comp,j)
            goto 678
          end if

          if(comp.eq.nmc) then
            liqtop=0.0d0
            do 878 i=1,nmc-1
              liqtop=liqtop+liqmasstr(i,j)*(xmolfrac(i,j)-xint(i,j))
 878        continue
            xint(comp,j)=xmolfrac(comp,j)+liqtop/liqmasstr(comp,j)
          else
            xint(comp,j)=(xmolfrac(comp,j)+gasmasstr(comp,j)/
     +           liqmasstr(comp,j)*ymolfrac(comp,j))/(gasmasstr(comp,j)/
     +           liqmasstr(comp,j)*keqcons(comp,j)+1.0d0)
          end if
          yint(comp,j)=keqcons(comp,j)*xint(comp,j)


 678      l=l+2
 766    continue

 82   continue


c  calculation of flux values for components

      na=liqmasstr(1,3)*intarea(3)*(xmolfrac(1,3)-xint(1,3))
      nb=liqmasstr(2,3)*intarea(3)*(xmolfrac(2,3)-xint(2,3))
      nc=liqmasstr(3,3)*intarea(3)*(xmolfrac(3,3)-xint(3,3))

c       normalization of mol fractions to 1.0
      call unif(numg,nmc,xmolfrac,xint,ymolfrac,yint)

      do 101 ii=1,numg
        temppre(ii)=ttk(ii)
        lrpre(ii)=liqrate(ii)
        grpre(ii)=gasrate(ii)
 101  continue

c       ************************************************************
c       step 6
c       new temperature calculations
      call newtempcal(xfeed,xref,xint,p,ttk)
      do 106 ii=1,numg
        blpre(ii)=liqhold(ii)
        bgpre(ii)=gashold(ii)
        mixhlpre(ii)=mixhl(ii)
        mixhvpre(ii)=mixhv(ii)
 106  continue

c       ********************************************************************
c       step 7
c       physical properties and holdups using new temperature and
c       mol fractions

      call propcal




      totlhold=0.0d0
      totghold=0.0d0
      do 246 ii=1,numg
        lhold(ii)=liqhold(ii)*crosarea*deltaz
        ghold(ii)=gashold(ii)*crosarea*deltaz
        totlhold=totlhold+liqhold(ii)
        totghold=totghold+gashold(ii)
 246  continue
      totlhold=totlhold/numg*colvol
      totghold=totghold/numg*colvol
```

```
      conhold=volconhold*volrdrum*mlden(numg)/lmixwme(numg)

c       condenser-reflux drum calculations
c       liquid reflux [kmol/hr]
      liqref=gasrate(numg)*crosarea
c     +     ymolfrac(3,numg)
      do 81 comp=1,nmc
        xref(comp)=ymolfrac(comp,numg)
 81   continue

c       calculation of new total liquid and vapour flow rates
c       throughout the column
c      if(option.eq.3) then
c         do 567 ii=1,numg
c            gasrate(ii)=vboil/crosarea
c            liqrate(ii)=liqref/crosarea
c567      continue
c      else
c       from matrix calculations
c      call enerstart(model,solen)
      call forbakenerstart(model,solen)

      kk=1
      do 107 ii=1,numg
        liqrate(ii)=solen(kk)
        gasrate(ii)=solen(kk+1)
        kk=kk+2
 107  continue
c      end if

      call propcal



c       ********************************************************
c       step 8

c       new reboiler holdup
      rebhold(1)=charge-totlhold-totghold-conhold

c       new reboiler composition
      top=0.0d0
      do 146 i=1,nmc
        totliq=0.0d0
        totvap=0.0d0
        totcon=0.0d0
        do 154 ii=1,numg
          totliq=totliq+lhold(ii)*xmolfrac(i,ii)
          totvap=totvap+ghold(ii)*ymolfrac(i,ii)
 154    continue
        totcon=totcon+conhold*xref(i)
        xfeed(i)=(charge*xb0(i)-totliq-totvap-totcon)/rebhold(1)
        top=top+xfeed(i)
 146  continue
      do 147 i=1,nmc
        xfeed(i)=xfeed(i)/top
 147  continue

      call rebtemp(presbot,xfeed,tstill,yboil)
      do 84 comp=1,nmc
        ymolfrac(comp,1)=yboil(comp)
 84   continue
      call  enthal(xfeed,yboil,tstill,presbot,
     +     rebhlnew,rebhvnew,v)

c       ********************************************************
c       heat input to the reboiler [kj/hr] or new vapour boilup rate [kmol/hr]

      if(decide.eq.1) then
        qreb=vboil*(rebhvnew-rebhlnew)-liqrate(1)*crosarea*
     +       (mixhl(1)-rebhlnew)
      end if

      if(decide.eq.2.or.decide.eq.3) then
        vboil=(qreb+liqrate(1)*crosarea*(mixhl(1)-rebhlnew))/
     +       (rebhvnew-rebhlnew)
      end if
c       condenser heat output qc[kj/hr]
      qc=gasrate(numg)*crosarea*(mixhv(numg)-mixhl(numg))


c       time increment
      time=time+deltatime

c      write(*,22222)time

c       ********************************************************
c       step 8
c       check norm of temperature if not repeat steps 2 through 8
c       difference from the previous values
      call normcal(temppre,ttk,norm)
      call matnormcal(xpre,xmolfrac,normx)
      call matnormcal(xintpre,xint,normxint)
      call matnormcal(ypre,ymolfrac,normy)
      call matnormcal(yintpre,yint,normyint)
      call normcal(lrpre,liqrate,normlr)
      call normcal(grpre,gasrate,normgr)

c      do 323 ii=1,numg
c        do 324 comp=1,nmc
c          write(*,5667)comp,xpre(comp,ii),xmolfrac(comp,ii),
c     +         ypre(comp,ii),ymolfrac(comp,ii),xintpre(comp,ii),
c     +         xint(comp,ii),yintpre(comp,ii),yint(comp,ii)
c 324    continue
c 323  continue
c 5667 format(i1,2x,f8.6,1x,f8.6,2x,f8.6,1x,f8.6,2x,f8.6,1x,f8.6,
c     +     2x,f8.6,1x,f8.6)

      write(28,2299)iter,norm,normlr,normgr,normx,normxint,
     +     normy,normyint
      write(*,2299)iter,norm,normlr,normgr,normx,normxint,
     +     normy,normyint


c       ********************************************************
c       step 11

      call presdropcal(totpresdrop)

      if(dabs(norm).le.1.0d-6) then
      write(30,8880)
 8880 format(/,6x,'Modelling Results',/,'Time [min]',2x,'T (0 cm)',1x,
     +      'T (42.5 cm)',1x,'T (85 cm)',1x,'T (127.5 cm)',1x,
     +     'T (170 cm)',1x,'DP (mm W)',1x,'presbot [atm]',1x,
     +     'prestop [atm]',1x,'Reb.Hold.[mol]',
     +     1x,'Qreb [kW]',1x,'Qcond [kW]',3x,'vboil [kmol/h]',1x,
     +     'Xfeed,mol %',4x,
     +     'y-top,mol%',4x,'Xreflux,mol %',2x,'Reflux=L/D',/)

      if(nmc.eq.2) then
      write(30,9990)time*60,tstill-273.15d0,ttk(int(0.425/deltaz))-
     +     273.15d0,ttk(int(0.85/deltaz))-273.15d0,
     +     ttk(int(1.275/deltaz))-273.15d0,ttk(int(1.7/deltaz))-
     +     273.15d0,totpresdrop,p(1),p(numg),rebhold(1)*1.0d3,qreb,qc,
     +     vboil,xfeed(1),
     +     ymolfrac(1,numg),
     +     xref(1)
      write(*,9990)time*60,tstill-273.15d0,ttk(int(0.425/deltaz))-
     +     273.15d0,ttk(int(0.85/deltaz))-273.15d0,
     +     ttk(int(1.275/deltaz))-273.15d0,ttk(int(1.7/deltaz))-
     +     273.15d0,totpresdrop,p(1),p(numg),rebhold(1)*1.0d3,qreb,qc,
     +     vboil,xfeed(1),
     +     ymolfrac(1,numg),
     +     xref(1)
 9990 format(f10.4,1x,5(f7.2,1x),3(f10.5,1x),f10.4,1x,f10.2,1x,f10.2,
```

```
     +     1x,f10.5,(1x,f10.8),(1x,f10.8),(1x,f10.8))
      else
      write(30,9991)time*60,tstill-273.15d0,ttk(int(0.425/deltaz))-
     +     273.15d0,ttk(int(0.85/deltaz))-273.15d0,
     +     ttk(int(1.275/deltaz))-273.15d0,ttk(int(1.7/deltaz))-
     +     273.15d0,totpresdrop,p(1),p(numg),rebhold(1)*1.0d3,qreb,qc,
     +     vboil,xfeed(1),xfeed(2),
     +     ymolfrac(1,numg),ymolfrac(2,numg),
     +     xref(1),xref(2)
      write(*,9991)time*60,tstill-273.15d0,ttk(int(0.425/deltaz))-
     +     273.15d0,ttk(int(0.85/deltaz))-273.15d0,
     +     ttk(int(1.275/deltaz))-273.15d0,ttk(int(1.7/deltaz))-
     +     273.15d0,totpresdrop,p(1),p(numg),rebhold(1)*1.0d3,qreb,qc,
     +     vboil,xfeed(1),xfeed(2),
     +     ymolfrac(1,numg),ymolfrac(2,numg),
     +     xref(1),xref(2)
 9991 format(f10.4,1x,5(f7.2,1x),3(f10.5,1x),f10.4,1x,f10.2,1x,f10.2,
     +     1x,f10.5,2(1x,f10.8),2(1x,f10.8),2(1x,f10.8))
      end if
c       record total reflux calculation results into totalref file
      write(19,7113)tstill,rebhold(1),conhold,qreb,qc,vboil
      write(19,7117)(xfeed(jj),jj=1,nmc)
      write(19,7117)(yboil(jj),jj=1,nmc)
      write(19,6789)((xmolfrac(i,j),j=1,numg),i=1,nmc)
      write(19,6789)((ymolfrac(i,j),j=1,numg),i=1,nmc)
      write(19,6789)((xint(i,j),j=1,numg),i=1,nmc)
      write(19,6789)((yint(i,j),j=1,numg),i=1,nmc)
      write(19,6789)(ttk(j),j=1,numg)
      write(19,6789)(liqrate(j),j=1,numg)
      write(19,6789)(gasrate(j),j=1,numg)

 7113 format(6(d22.16,1x))
 7117 format(3(d22.16,1x))
 6789 format(30(d22.16,1x))
c       ----------------------------------------------------------------
c       return to main program
      call  out(prog,time,iter,iteres,morvol,totdistil,
     +     xdisaverage,distil,reflux,qreb,qc,norm,totlhold,totghold)
      goto 1001
      end if

      if(mod(iter,yaztot).eq.0) then
        call out(prog,time,iter,iteres,morvol,totdistil,
     +      xdisaverage,distil,reflux,qreb,qc,norm,totlhold,totghold)
      end if
c       ----------------------------------------------------------------
c       relaxation step
      do 203 j=1,numg
        do 24 comp=1,nmc
          xmolfrac(comp,j)=(1.0d0-relaxfac)*xpre(comp,j)+
     +          relaxfac*xmolfrac(comp,j)
          xint(comp,j)=(1.0d0-relaxfac)*xintpre(comp,j)+
     +          relaxfac*xint(comp,j)
          ymolfrac(comp,j)=(1.0d0-relaxfac)*ypre(comp,j)+
     +          relaxfac*ymolfrac(comp,j)
          yint(comp,j)=(1.0d0-relaxfac)*yintpre(comp,j)+
     +          relaxfac*yint(comp,j)
 24     continue
        ttk(j)=(1.0d0-relaxfac)*temppre(j)+relaxfac*ttk(j)
        liqrate(j)=(1.0d0-relaxfac)*lrpre(j)+relaxfac*liqrate(j)
        gasrate(j)=(1.0d0-relaxfac)*grpre(j)+relaxfac*gasrate(j)
 203  continue

      iter=iter+1
      goto 1000

c       ----------------------------------------------------------------

 2233 format(/,/,'iteration=',i5,/,'norm of temp.=',e15.10,/,
     +     'norm of liqrate=',
     +     e15.10,/,'norm of gasrate=',e15.10,/,'norm of x=',
     +     e15.10,/,'norm of xint=',e15.10,/,'norm of y=',e15.10,/,
     +     'norm of yint=',e15.10,/)
 2299 format(i5,3x,f15.10,3x,f15.10,3x,f15.10,3x,f15.10,
     +     3x,f15.10,3x,f15.10,3x,f15.10)


 1110 format(/,'reboiler mixture',/,/,'still temperature [c]=',
     +     f15.9,/)
 1111 format(1x,'component',11x,'x,mol%',5x,'yboil,mol%')
 1112 format(1x,'---------',11x,'------',5x,'----------')
 1113 format(1x,a15,3x,f10.8,3x,f10.8)
 778  format(/,'condenser heat duty [kj/hr]=',f15.9,/)
 787  format(/,'reboiler heat duty [kj/hr]=',f15.9,/)
 9999 format(/,'results using central finite difference method',/,/)
 8889 format(/,'component=',a15,/,/,1x,'stage',7x,'x',12x,
     +     'xint',10x,'y',12x,'yint')
 8890 format(1x,'-----',3x,3('-----------',5x),'----------')
 9997 format(i4,4f15.9)

c       ----------------------------------------------------------------

c       close files and return to main program
 1001 close (unit=24)
      close (unit=25)
      close (unit=26)
      close (unit=27)
      close (unit=28)
      close (unit=19)
      return
      end



      subroutine rebtemp(p,xfeed,tstill,yboil)
c*****************************************************************
c     new reboiler temperature from bubble point calculation
c     vapour composition of the reboiler mixture
c*****************************************************************

      implicit double precision (a-z)
      integer numg,nmc,comp
      include 'param.f'
      common/phis/tc(1:nmc),tbp(1:nmc),pc(1:nmc),w(1:nmc),vc(1:nmc)
      dimension xfeed(1:nmc),yboil(1:nmc)
      dimension keq(1:nmc)

c ^Iassumption for temperature of the still [k]

      temp=tbp(1)+273.15d0
      step=0.01d0
      iter=1
 62   call kval(1,temp,p,xfeed,actliq,keq)
      call tempfonk(xfeed,keq,functemp)
      if (dabs(functemp).le.0.1d-10) then
        tstill=temp
        do 212 comp=1,nmc
          yboil(comp)=keq(comp)*xfeed(comp)
 212    continue
        goto 63
      else
c       derivative function
        tempold=temp
        funcold=functemp
        temp=temp+step
        call kval(1,temp,p,xfeed,actliq,keq)
        call tempfonk(xfeed,keq,functemp)
        temp=tempold-funcold/((functemp-funcold)/step)
        iter=iter+1
      end if
      goto 62
 63   return
      end

      subroutine assign(nmc,numg,proper)
c*****************************************************************
```

```
c     it could be used for constant property assumption case
c     to assign values of the first node in the packed section to
c     the other nodes' values
c**********************************************************************

      implicit double precision (a-z)
      integer nmc,numg,i,j
      dimension proper(1:nmc,1:numg)
      do 1 i=1,nmc
        do 2 j=1,numg
          proper(i,j)=proper(i,1)
 2      continue
 1    continue
      return
      end

      subroutine unif(numg,nmc,xmf,xi,ymf,yi)
c**********************************************************************
c     it sums the mol fraction values for the liquid and vapour
c     phases to 1.0 and calculates the corrected mol fraction values
c**********************************************************************

      implicit double precision (a-z)
      integer nmc,numg,ii,comp
      dimension xmf(1:nmc,1:numg),xi(1:nmc,1:numg)
      dimension yi(1:nmc,1:numg),ymf(1:nmc,1:numg)
      do 553 ii=1,numg
        topx=0.0d0
        topy=0.0d0
        topxi=0.0d0
        topyi=0.0d0

        do 554 comp=1,nmc
          topx=topx+xmf(comp,ii)
          topxi=topxi+xi(comp,ii)
          topy=topy+ymf(comp,ii)
          topyi=topyi+yi(comp,ii)
 554    continue

        do 556 comp=1,nmc
          if (topx.ne.0.0d0) then
            xmf(comp,ii)=xmf(comp,ii)/topx
          endif

          if (topy.ne.0.0d0) then
            ymf(comp,ii)=ymf(comp,ii)/topy
          endif

          if (topyi.ne.0.0d0) then
            yi(comp,ii)=yi(comp,ii)/topyi
          endif

          if (topxi.ne.0.0d0) then
            xi(comp,ii)=xi(comp,ii)/topxi
          endif
 556    continue

 553  continue
      return
      end

      subroutine newtempcal(xf,xr,xi,pres,tempar)
c**********************************************************************
c     temperature calculations of the nodes taken from the packed
c     section by bubble point calculations
c**********************************************************************

      implicit double precision (a-z)
      integer numg,nmc,ii,comp
      include 'param.f'
      common/phis/tc(1:nmc),tbp(1:nmc),pc(1:nmc),w(1:nmc),vc(1:nmc)
      dimension tempar(1:numg),pres(1:numg),xi(1:nmc,1:numg)
      dimension xintges(1:nmc),keq(1:nmc),xf(1:nmc),xr(1:nmc)

c       column interfacial temperatures via bubble point calculations
      step=0.001d0
      do 85 ii=1,numg
        p=pres(ii)
c        temp1=tbp(1)+273.15d0
c        temp2=tbp(nmc)+273.15d0
        temp1=tempar(ii)-5.0d0
        temp2=tempar(ii)+5.0d0
        topxintges=0.0d0
        do 52 comp=1,nmc
          xintges(comp)=xi(comp,ii)
          if(ii.eq.1) then
            xintges(comp)=xf(comp)
          end if
          if(ii.eq.numg) then
            xintges(comp)=xr(comp)
          end if
          topxintges=topxintges+xintges(comp)
 52     continue
        if (topxintges.eq.0.0d0) goto 85
 172    tempar(ii)=temp1
        call kval(1,temp1,p,xintges,actliq,keq)
        call tempfonk(xintges,keq,functemp1)
        tempar(ii)=temp2
        call kval(1,temp2,p,xintges,actliq,keq)
        call tempfonk(xintges,keq,functemp2)
c     +       temp1-273.15,temp2-273.15
        if (functemp1*functemp2.ge.0.0d0) then
          temp1=temp1-1.0d0
          temp2=temp2+1.0d0
          goto 172
        end if
        fl=min(functemp1,functemp2)
        fh=max(functemp1,functemp2)
        if(fl.eq.functemp1) then
          xl=temp1
          xh=temp2
        else
          xl=temp2
          xh=temp1
        end if
        rtsafe=0.5d0*(temp1+temp2)
        dxold=dabs(temp2-temp1)
        dx=dxold
        tempar(ii)=rtsafe
        call kval(1,rtsafe,p,xintges,actliq,keq)
        call tempfonk(xintges,keq,f)
        call kval(1,rtsafe+step,p,xintges,actliq,keq)
        call tempfunctur(xintges,keq,step,f,df)
        do 157 j=1,50
          if ((((rtsafe-xh)*df-f)*((rtsafe-xl)*df-f)).ge.0.0d0.
     +         or.(dabs(2.0d0*f)).gt.(dabs(dxold*df))) then
            dxold=dx
            dx=0.5d0*(xh-xl)
            rtsafe=xl+dx
            if (xl.eq.rtsafe) goto 159
          else
            dxold=dx
            dx=f/df
            temp=rtsafe
            rtsafe=rtsafe-dx
            if (temp.eq.rtsafe) goto 159
          end if
          if (dabs(dx).lt.1.0d-10) goto 159
          tempar(ii)=rtsafe
          call kval(1,rtsafe,p,xintges,actliq,keq)
          call tempfonk(xintges,keq,f)
          call kval(1,rtsafe+step,p,xintges,actliq,keq)
          call tempfunctur(xintges,keq,step,f,df)
          if (f.lt.0.0d0) then
            xl=rtsafe
            fl=f
          else
```

```
2337              xh=rtsafe
2338              fh=f
2339            end if
2340   157    continue
2341   159    tempar(ii)=rtsafe
2342   85   continue
2343        return
2344        end
2345
2346        subroutine normcal(temppre,ttk,norm)
2347   c*********************************************************************
2348   c     norm of the difference between two subsequent calculated
2349   c     values (temperatures, molar fluxes of the nodes in the packed
2350   c     section) for total reflux condition
2351   c*********************************************************************
2352
2353        implicit double precision (a-z)
2354        integer i,numg
2355        include 'param.f'
2356        dimension temppre(1:numg),ttk(1:numg)
2357        norm=0.0d0
2358        do 1 i=1,numg
2359          norm=norm+(ttk(i)-temppre(i))**2.0d0
2360    1     continue
2361        norm=norm**0.5d0
2362        return
2363        end
2364
2365        subroutine matnormcal(xpre,xmolfrac,norm)
2366   c*********************************************************************
2367   c     norm calculations for molar fractions
2368   c*********************************************************************
2369
2370        implicit double precision (a-z)
2371        integer i,numg,nmc,j
2372        include 'param.f'
2373        dimension xpre(1:nmc,1:numg),xmolfrac(1:nmc,1:numg)
2374        norm=0.0d0
2375        do 2 j=1,nmc
2376          do 1 i=1,numg
2377            norm=norm+(xmolfrac(j,i)-xpre(j,i))**2.0d0
2378    1       continue
2379    2     continue
2380        norm=norm**0.5d0
2381        return
2382        end
2383
2384        subroutine out(prog,time,iter,iteres,morvol,totdistil,
2385       +      xdisaverage,distil,reflux,qreb,qc,norm,totlhold,totghold)
2386   c *********************************************************************
2387   c     print the results into the files with given frequency
2388   c     prog=1 for total reflux calculations
2389   c     prog=2 for product period
2390   c     different output files are opened for each period
2391   c *********************************************************************
2392
2393        implicit double precision (a-z)
2394        integer nmc,comp,numg,yaztot,yazpro,option
2395        integer ii,iter,iteres,jj,compon,decide,prog,model
2396        integer nmol,nphy,nflow,nerror,morvol
2397        include 'param.f'
2398        include 'comblok.f'
2399
2400        dimension whereinpac(1:numg),reflux(1:nmc)
2401        character *15 namecomp(1:nmc)
2402
2403        if(prog.eq.1) then
2404          nmol=24
2405          nphy=25
2406          nflow=26
2407          nerror=28
2408        end if
2409
2410        if(prog.eq.2) then
2411          nmol=14
2412          nphy=15
2413          nflow=16
2414          nerror=18
2415        end if
2416        if(prog.eq.2) then
2417          write(20,7770)time*60.0d0,iter,iteres,namecomp(morvol),
2418       +        reflux(morvol),distil,totdistil,xdisaverage,liqref,
2419       +        xref(morvol),vboil
2420          write(20,1110)(tstill-273.15d0)
2421          write(20,1111)
2422          write(20,1112)
2423          do 2651 jj=1,nmc
2424            write(20,1113)namecomp(jj),xfeed(jj),yboil(jj)
2425   2651     continue
2426          write(20,1117)rebhold(1),conhold,totlhold,totghold,qreb,qc
2427        end if
2428   7770  format(/,/,'time [min]=',f15.5,/,'time iteration step=',i5,/,
2429       +      'corrector step=',i5,/,'product taken=',a15,/,
2430       +      'reflux ratio (L/D)=',f10.5,/,
2431       +      'distillate rate [kmol/h]=',f15.10,/,
2432       +      'amount of product in the collector [kmol]=',f15.10,/,
2433       +      'product composition in the collector=',f15.10,/,
2434       +      'reflux rate [kmol/h]=',f15.6,/,
2435       +      'reflux composition=',f15.10,/,
2436       +       'boilup rate [kmol/h]=',f15.6)
2437
2438        write(nmol,22222)time
2439        write(nphy,22222)time
2440        write(nflow,22222)time
2441   22222 format(/,'time=',f15.10)
2442        write(nmol,33333)iter
2443        write(nphy,33333)iter
2444        write(nflow,33333)iter
2445   33333 format('iteration no=',i5)
2446        whereinpac(1)=0.0d0
2447        do 601 ii=2,numg
2448          whereinpac(ii)=whereinpac(ii-1)+deltaz
2449    601   continue
2450        do 199 comp=1,nmc
2451          write(nphy,44440)namecomp(comp)
2452          write(nphy,44444)
2453          write(nphy,44445)
2454          write(nphy,44446)
2455          do 201 ii=1,numg
2456            write(nphy,54447)ii,whereinpac(ii),keqcons(comp,ii),
2457       +        liqden(comp,ii),gasden(comp,ii),liqvis(comp,ii),
2458       +        gasvis(comp,ii)
2459    201     continue
2460
2461          write(nphy,54444)
2462          write(nphy,54445)
2463          write(nphy,54446)
2464          do 2201 ii=1,numg
2465            write(nphy,44447)ii,whereinpac(ii),mldif(comp,ii),
2466       +        mgdif(comp,ii),liqmasstr(comp,ii),gasmasstr(comp,ii)
2467    2201    continue
2468    199   continue
2469
2470        write(nphy,44443)
2471        do 6012 ii=1,numg
2472          write(nphy,44449)ii,whereinpac(ii),ttk(ii)-273.15,p(ii),
2473       +      laxdis(ii),gaxdis(ii)
2474    6012 continue
2475
2476        write(nphy,8881)
2477        write(nphy,8882)
2478        do 3016 ii=1,numg
2479          write(nphy,8883)ii,whereinpac(ii),mixhl(ii),mixhv(ii)
2480    3016 continue
2481
2482    8881 format(/,/,'node',5x,' Z [m] ',10x,'Liq.Enthalpy [kJ/kmol]',
```

```
     +     5x,'Vap.Enthalpy [kJ/kmol]')
 8882 format('----',3x,'----------',7x,23('-'),4x,23('-'))
 8883 format(i4,3x,f10.7,13x,f12.4,10x,f12.4)

      write(nphy,74444)
      write(nphy,74445)
      write(nphy,74446)
      do 3012 ii=1,numg
        write(nphy,74447)ii,whereinpac(ii),mlden(ii),mgden(ii),
     +       mlvis(ii),mgvis(ii)
 3012 continue

      write(nphy,84444)
      write(nphy,84445)
      write(nphy,84446)
      do 3013 ii=1,numg
        write(nphy,84447)ii,whereinpac(ii),mlst(ii),intarea(ii),
     +       lhold(ii),ghold(ii)
 3013 continue

      do 76 comp=1,nmc
        write(nmol,8889)namecomp(comp)
        write(nmol,8890)
        do 772 ii=1,numg
          write (nmol,9997) ii,whereinpac(ii),xmolfrac(comp,ii),
     +         xint(comp,ii),yint(comp,ii),ymolfrac(comp,ii)
  772   continue
   76   continue

      write (nflow,7778)
      write (nflow,6667)
      do 703 ii=1,numg
        write (nflow,9998) ii,whereinpac(ii),ttk(ii)-273.15d0,
     +       liqrate(ii),lrate(ii),gasrate(ii),grate(ii)
  703  continue


c       ------------------------------------------------------------
 1134 format (/,'error=',f15.9,' iteration=',i4)

 1117 format(/,'rebhold [kmol]=',f15.9,/,'conhold [kmol]=',f15.9,
     +     /,'tot. liq. hold. in packed section [kmol]=',f15.9,/,
     +     'tot. gas. hold. in packed section [kmol]=',f15.9,/,
     +     'qreb [kj/hr]=',f20.9,/,'qcond [kj/hr]=',f20.9,/)

 1110 format(/,'reboiler mixture',/,/,'still temperature [c]=',
     +     f15.9,/)
 1111 format(1x,'component',11x,'x,mol%',5x,'yboil,mol%')
 1112 format(1x,'---------',11x,'------',5x,'----------')
 1113 format(1x,a15,3x,f10.8,3x,f10.8)
64444 format(/,1x,'node',4x,'liqrate',8x,'gasrate',9x,'liqrate',
     +     8x,'gasrate')
64445 format(16x,'[kmol/m2 hr]',20x,'[kg/m2 hr]')
64446 format(1x,'-----',2x,4('-----------',4x))
 778  format(/,'condenser heat duty [kw]=',f15.9,/)
 787  format(/,'reboiler heat duty [kw]=',f15.9,/)
44440 format(2(/),'component=',a15,/)
44444 format(/,'node',5x,'Z [m]',10x,'k',8x,'liqden',6x,'gasden',
     +     6x,'liqvis',6x,'gasvis')
44445 format(38x,'[kg/m3]',18x,'[kg /m hr]')
44446 format('-----',2x,'----------',3x,5('---------',3x))
44447 format(i4,2x,f10.7,3x,f10.7,4x,f10.7,5x,f10.7,5x,f10.7,5x,f10.7)
54447 format(i4,2x,f10.7,3x,f10.7,2x,f10.5,2x,f10.7,2x,f10.7,2x,f10.7)
54444 format(/,/,'node',5x,' Z [m] ',5x,'liqdiff',10x,'gasdiff',
     +     10x,'kx',12x,'ky')
54445 format(28x,'[m2/hr]',22x,'[kmol/m2 hr]')
54446 format('-----',2x,'----------',3x,4('-----------',4x))
44443 format(/,/,30x,'mixture properties',/,/,'node',9x,
     +     'Z [m]',6x,'Temp [C]',
     +     3x,'pres [atm]',2x,'Liq.ax.Dis.',4x,
     +     'Vap.ax.Dis.[m2/h]',/,'----',6x,'----------',2x,
     +     '----------',3x,'----------',3x,'----------',3x,'----------')
44449 format(i4,5x,f10.7,3x,f10.5,3x,f10.8,3x,f10.7,3x,f10.5)

74444 format(/,'node',5x,' Z [m] ',5x,
     +     'liqden',8x,'gasden',9x,'liqvis',
     +     9x,'gasvis')
74445 format(27x,'[kg/m3]',23x,'[kg /m hr]')
74446 format('-----',2x,'----------',2x,4('-----------',4x))
74447 format(i3,2x,f10.7,4f15.9)
84444 format(/,/,'node',5x,' Z [m] ',5x,'liqsurten',6x,'intarea',7x,
     +     'liqholdup',6x,'gasholdup')
84445 format(25x,'[n/m]',8x,'[m2/m3]',17x,'[kmol]')
84446 format('-----',2x,'----------',2x,4('-----------',4x))
84447 format(i3,3x,f10.7,4f15.9)
 9889 format(/,'derivative results',/,1x,'node',7x,'xp',
     +     12x,'xintp',14x,'yp')
 9890 format(1x,'-----',3x,2('----------',5x),'----------')
 8889 format(/,'component=',a15,/,/,'node',5x,' Z [m] ',8x,'x',12x,
     +     'xint',10x,'yint',12x,'y')
 8890 format('-----',2x,'---------',4(3x,'-----------'))
 9997 format(i4,2x,f10.7,4(3x,f11.9))
 7778 format (/,/,
     +     'node',6x,'Z',9x,'temp',11x,'L',9x,
     +     'L',12x,'V',9x,'V',/,9x,'[m]',9x,'[C]',7x,'[kmol/m2 h]',
     +     1x,'[kg/m2 h]',2x,'[kmol/m2 h]',1x,'[kg/m2 h]')
 6668 format (12x,'[c]',17x,'[kj/kmol]',20x,'[kmol/m2 hr]')
 6667 format ('-----',2x,'---------',3x,9('-'),2(2x,22('-')))
 9998 format(i4,2x,f10.7,2x,f10.5,2x,f10.5,2x,f10.5,2x,f10.5,2x,f10.5,2x,f10.5)
      return
      end

      subroutine masstransfer(model,sol)
c ********************************************************************
c     it solves the steady state mass balance equations for x,y
c     using central finite difference method
c ********************************************************************

      implicit double precision (a-z)
      integer numg,j,i,model,nmc,comp
      include 'param.f'
      common/disperblok/laxdis(1:numg),gaxdis(1:numg)
      common/holdupblok/volrdrum,volconhold,preconhold,
     +     conhold,rebhold(1),liqhold(1:numg),gashold(1:numg)
      common/flowrate/liqrate(1:numg),gasrate(1:numg)
      common/mastran/liqmasstr(1:nmc,1:numg),gasmasstr(1:nmc,1:numg)
      common/prehold/blpre(1:numg),bgpre(1:numg)
      common/molfrac/xmolfrac(1:nmc,1:numg),ymolfrac(1:nmc,1:numg)
      common/premolfrac/xpre(1:nmc,1:numg),ypre(1:nmc,1:numg)
      common/prop1a/intarea(1:numg),keqcons(1:nmc,1:numg)
      common/start1/deltatime,deltaz,comp
      common/boundary/yboil(1:nmc),xref(1:nmc)
      common/parastudy/alfaky,alfakx,beta1,beta2

      dimension zmat(2*numg,2*numg),gval(2*numg),sol(2*numg)
      do 11 i=1,2*numg
        do 22 j=1,2*numg
          zmat(i,j)=0.0d0
 22     continue
        gval(i)=0.0d0
        sol(i)=0.0d0
 11   continue

c       reboiler-column interface

      zmat(1,1)=-3.0d0

      zmat(1,3)=4.0d0

      zmat(1,5)=-1.0d0


      zmat(2,2)=1.0d0

      gval(2)=yboil(comp)

c       column-condenser interface
```

```
      zmat(2*numg-1,2*numg-1)=1.0d0

      gval(2*numg-1)=xref(comp)


      zmat(2*numg,2*numg-4)=1.0d0

      zmat(2*numg,2*numg-2)=-4.0d0

      zmat(2*numg,2*numg)=3.0d0


c       column section
      do 93 j=2,numg-1

        coff=gasmasstr(comp,j)*intarea(j)/
     +       (gasmasstr(comp,j)/liqmasstr(comp,j)*keqcons(comp,j)+1.0d0)

        zmat(j*2-1,(j-1)*2-1)=-liqrate(j)/(2.0d0*deltaz)

        zmat(j*2-1,2*j-1)=-coff*keqcons(comp,j)

        if(j.ne.numg-1) then
          zmat(j*2-1,(j+1)*2-1)=liqrate(j)/(2.0d0*deltaz)
        end if

        zmat(j*2-1,2*j)=coff


        if(j.eq.numg-1) then
          gval(j*2-1)=-xref(comp)*(liqrate(j)/(2.0d0*deltaz))
        end if

        zmat(2*j,2*j-1)=coff*keqcons(comp,j)

        if(j.ne.2) then
          zmat(j*2,(j-1)*2)=gasrate(j)/(2.0d0*deltaz)
        end if

        if(j.eq.2) then
          gval(j*2+2)=-yboil(comp)*(gasrate(j)/(2.0d0*deltaz))
        end if

        zmat(j*2,2*j)=-coff


        zmat(j*2,(j+1)*2)=-gasrate(j)/(2.0d0*deltaz)


        if(model.eq.2) then
          zmat(j*2-1,(j-1)*2-1)=zmat(j*2-1,(j-1)*2-1)
     +         +laxdis(j)*liqhold(j)/(deltaz**2.0d0)

          zmat(j*2-1,2*j-1)=zmat(j*2-1,2*j-1)
     +         -2.0d0*laxdis(j)*liqhold(j)/(deltaz**2.0d0)

          if(j.ne.numg-1) then
            zmat(j*2-1,(j+1)*2-1)=zmat(j*2-1,(j+1)*2-1)
     +         +laxdis(j)*liqhold(j)/(deltaz**2.0d0)
          end if


          if(j.eq.numg-1) then
            gval(j*2-1)=gval(j*2-1)
     +           -xref(comp)*(laxdis(j)*liqhold(j)/(deltaz**2.0d0))
          end if


          if(j.ne.2) then
            zmat(j*2,(j-1)*2)=zmat(j*2,(j-1)*2)
     +           +gaxdis(j)*gashold(j)/(deltaz**2.0d0)
          end if


            if(j.eq.2) then
            gval(j*2+2)=gval(j*2+2)
     +           -yboil(comp)*(gaxdis(j)*gashold(j)/(deltaz**2.0d0))
          end if


          zmat(j*2,2*j)=zmat(j*2,2*j)
     +         -2.0d0*gaxdis(j)*gashold(j)/(deltaz**2.0d0)


          zmat(j*2,(j+1)*2)=zmat(j*2,(j+1)*2)
     +         +gaxdis(j)*gashold(j)/(deltaz**2.0d0)
        end if
 93     continue

        call gauss(2*numg,numg,zmat,gval,sol)

c         kayit=0
c         do 90 i=1,2*numg
c         do 91 j=1,2*numg
c         if(zmat(i,j).ne.0.0d0) then
c         kayit=kayit+1
c         write(2,2345)kayit,i,j,zmat(i,j)
c         write(*,2345)kayit,i,j,zmat(i,j)
c         end if
c91       continue
c90       continue
c         write(2,*)
c         write(3,2346)(gval(i),i=1,2*numg)
c         write(*,2346)(gval(i),i=1,2*numg)
c         write(3,2347)(sol(i),i=1,2*numg)
c2345     format(i5,i5,i5,1x,(d22.15))
c2346     format('gval=',30d22.15)
c2347     format('solu=',30d22.15)
        return
        end


        subroutine masstrforward(model,sol)
c ****************************************************************
c     it solves steady state mass balance equations for x,y
c     using forward difference for liquid phase and bacward
c     difference for vapour phase
c ****************************************************************

        implicit double precision (a-z)
        integer numg,kil,model,j,i,nmc,comp
        include 'param.f'

        common/molfrac/xmolfrac(1:nmc,1:numg),ymolfrac(1:nmc,1:numg)
        common/disperblok/laxdis(1:numg),gaxdis(1:numg)
        common/holdupblok/volrdrum,volconhold,preconhold,
     +      conhold,rebhold(1),liqhold(1:numg),gashold(1:numg)
        common/flowrate/liqrate(1:numg),gasrate(1:numg)
        common/mastran/liqmasstr(1:nmc,1:numg),gasmasstr(1:nmc,1:numg)
        common/propla/intarea(1:numg),keqcons(1:nmc,1:numg)
        common/start1/deltatime,deltaz,comp
        common/boundary/yboil(1:nmc),xref(1:nmc)
        common/parastudy/alfaky,alfakx,beta1,beta2

        dimension zmat(2*numg,2*numg),gval(2*numg),sol(2*numg)

        do 11 i=1,2*numg
          do 22 j=1,2*numg
            zmat(i,j)=0.0d0
 22       continue
          gval(i)=0.0d0
          sol(i)=0.0d0
 11     continue


c       reboiler-column interface

        zmat(1,1)=-3.0d0
```

```
      zmat(1,3)=4.0d0
      zmat(1,5)=-1.0d0

      zmat(2,2)=1.0d0

      gval(2)=yboil(comp)

c       column-condenser interface
      zmat(2*numg-1,2*numg-1)=1.0d0

      gval(2*numg-1)=xref(comp)


      zmat(2*numg,2*numg-4)=1.0d0
      zmat(2*numg,2*numg-2)=-4.0d0
      zmat(2*numg,2*numg)=3.0d0


c       column section
      do 93 j=2,numg-1

        coff=gasmasstr(comp,j)*intarea(j)/
     +       (gasmasstr(comp,j)/liqmasstr(comp,j)*keqcons(comp,j)+1.0d0)


        zmat(2*j-1,2*(j)-1)=-liqrate(j)/deltaz-coff*keqcons(comp,j)

        if(j.ne.numg-1) then
          zmat(2*j-1,2*(j+1)-1)=liqrate(j)/(deltaz)
        end if

        zmat(2*j-1,2*j)=coff


        if(j.eq.numg-1) then
          gval(2*j-1)=-xref(comp)*(liqrate(j)/(deltaz))
        end if

        if(j.ne.2) then
          zmat(2*j,2*(j-1))=gasrate(j)/(deltaz)
        end if

        if(j.eq.2) then
          gval(2*j)=-yboil(comp)*(gasrate(j)/(deltaz))
        end if

        zmat(2*j,2*j-1)=coff*keqcons(comp,j)


        zmat(2*j,2*j)=-coff-gasrate(j)/deltaz


        if(model.eq.2) then
          zmat(2*j-1,2*(j-1)-1)=zmat(2*j-1,2*(j-1)-1)
     +         +laxdis(j)*liqhold(j)/(deltaz**2.0d0)

          zmat(2*j-1,2*(j)-1)=zmat(2*j-1,2*(j)-1)
     +         -2.0d0*laxdis(j)*liqhold(j)/(deltaz**2.0d0)

          if(j.ne.numg-1) then
            zmat(2*j-1,2*(j+1)-1)=zmat(2*j-1,2*(j+1)-1)
     +           +laxdis(j)*liqhold(j)/(deltaz**2.0d0)
          end if

          if(j.eq.numg-1) then
            gval(2*j-1)=gval(2*j-1)
     +           -(laxdis(j)*liqhold(j)/(deltaz**2.0d0))*xref(comp)
          end if

          if(j.ne.2) then
            zmat(2*j,2*(j-1))=zmat(2*j,2*(j-1))
     +           +gaxdis(j)*gashold(j)/(deltaz**2.0d0)
          end if


          if(comp.eq.2) then
            gval(2*j)=gval(2*j)
     +           -(gaxdis(j)*gashold(j)/(deltaz**2.0d0))*yboil(comp)
          end if


          zmat(2*j,2*j)=zmat(2*j,2*j)
     +         -2.0d0*gaxdis(j)*gashold(j)/(deltaz**2.0d0)

          zmat(2*j,2*(j+1))=zmat(2*j,2*(j+1))+
     +         gaxdis(j)*gashold(j)/(deltaz**2.0d0)

        end if
        if(comp.eq.nmc) then
          flux=0.0d0
          do 99 kil=1,nmc-1
            coff=gasmasstr(kil,j)*intarea(j)/
     +           (gasmasstr(kil,j)/liqmasstr(kil,j)*
     +           keqcons(kil,j)+1.0d0)
            flux=flux-coff*(keqcons(kil,j)*xmolfrac(kil,j)-
     +           ymolfrac(kil,j))
 99       continue
          gval(2*j-1)=gval(2*j-1)+flux
          gval(2*j)=gval(2*j)-flux
        end if

 93     continue

c         do 90 i=1,2*numg
c         do 91 j=1,2*numg
c         if(zmat(i,j).ne.0.0d0) then
c         write(*,2345)i,j,zmat(i,j)
c         end if
c91       continue
c90       continue
c         write(2,*)
c         write(*,2346)(gval(i),i=1,2*numg)
c         write(3,2347)(sol(i),i=1,2*numg)
c2345     format(i5,i5,1x,e20.10)
c2346     format('gval=',2e20.10)

      call gauss(2*numg,numg,zmat,gval,sol)

      return
      end



      subroutine hgas(temp,icom,gasent)
c ********************************************************************
c       this routine calculates the enthalpy of ideal gases
c       by api method 7a1.1, (kj/kmol)
c       for i.component , j. stage
c ********************************************************************

      implicit double precision (a-z)
      integer icom,nmc,numg
      include 'param.f'
      common/identcons/a1(1:nmc),b1(1:nmc),c1(1:nmc),d1(1:nmc),e1(1:nmc)
      common/mixmolw/lmixwme(1:numg),vmixwme(1:numg),wme(1:nmc)
c       temp is given in [C] and tempe is in [R]
      tempe=(temp*9.0d0/5.0d0+32.0d0+459.67d0)/100.0d0
      gasent=(a1(icom)*tempe+b1(icom)*tempe**2.0d0+c1(icom)*
     +     tempe**3.0d0+d1(icom)*1.0d0/tempe+e1(icom))*wme(icom)*2.326d0
c       gasent is calculated in [kj/kmol]
      return
      end


      subroutine enthal(x,y,temp,pres,
     +     mixhliq,mixhvap,vv)
c ********************************************************************
```

```
c       this routine calculates liquid and vapour mixture enthalpies
c       calculation of enthalpy departure function by srk
c       it receives the temperature of the stage in [K]
c       pressure in [atm], liquid and vapour mol fraction values
c       calls hgas subroutine for ideal gas enthalpy calculations
c       returns liquid and vapour mixture enthalpies in [kj/kmol]
c ****************************************************************

      implicit double precision (a-z)
      integer comp,choose,kk,chose,nmc,numg
      include 'param.f'
      common/consbl/gamaa,gamab,coneq(19,4)
      common/phis/tc(1:nmc),tbp(1:nmc),pc(1:nmc),
     +     w(1:nmc),vc(1:nmc)
      common/mixmolw/lmixwme(1:numg),vmixwme(1:numg),wme(1:nmc)
      dimension x(1:nmc),y(1:nmc)
      dimension v(2)
      dimension vv(1:nmc),pr(1:nmc),tr(1:nmc)
      dimension hmho(2),z(3)
      tophl=0.0d0
      tophv=0.0d0
c       loop for components
c       R is in [cm3 atm/mol k]
c       r=82.057d0
      do 7 comp=1,nmc
        tr(comp)=temp/tc(comp)
        pr(comp)=pres/pc(comp)
^Isb=gamab*82.057d0*tc(comp)/pc(comp)
^If666=0.48d0+1.574d0*w(comp)-0.176d0*w(comp)**2.0d0
^If=1.0d0/tr(comp)*(1.0d0+f666*(1.0d0-tr(comp)**0.5d0))**2.0d0
^Ialfa=(1.0d0+f666*(1.0d0-tr(comp)**0.5d0))**2.0d0
^Iac=gamaa*82.057d0**2.0d0*tc(comp)**2.0d0/pc(comp)
^Iat=ac*alfa
^Icapa=at*pres/(82.057d0**2.0d0*temp**2.0d0)
^Ib=gamab*82.057d0*tc(comp)/pc(comp)
^Icapb=b*pres/(82.057d0*temp)
c       finding roots by bairstow method
 ^Icall roots(capa,capb,z)
^Ichoose=1
^Ichose=1
^Ikk=1
 120    if (z(choose).eq.0.0d0) then
          choose=1
^Iend if
        v(kk)=z(choose)*82.057d0*temp/pres
^Iif(chose.eq.1) then
          vv(comp)=v(kk)
^Iend if
^Ibeta=gamaa/gamab*b*f
^Ipo=0.987d0
^Ivo=82.057d0*temp/po
c^Ir=8.3144d0 [j/mol k]
^Iamao1=(beta/b*dlog((v(kk)+b)/v(kk))+dlog(v(kk)/vo))
^Iamao=8.3144d0*temp*(-dlog((v(kk)-b)/v(kk))-amao1)
^Igama=(gamaa/gamab*b*f**0.5d0*f666)
^Ismso1=gama/b*dlog((v(kk)+b)/v(kk))-dlog(v(kk)/vo)
^Ismso=8.3144d0*(dlog((v(kk)-b)/v(kk))-smso1)
^I
        hmho(kk)=amao+temp*smso+8.3144d0*temp*(z(choose)-1.0d0)
^Iif (choose.eq.1.and.kk.eq.1) then
          choose=3
          kk=2
          goto 120
^Iend if
c       calculation of ideal gas enthalpy by api method, kj/kmol
        call hgas((temp-273.15d0),comp,ent)
        entpv=ent+hmho(2)
 ^Ientpl=ent+hmho(1)
^Itophv=tophv+entpv*y(comp)
^Itophl=tophl+entpl*x(comp)
c       start for other component
 7    continue
      mixhliq=(tophl)
      mixhvap=(tophv)
      return
      end



      subroutine kval(stage,temp,pres,x,actliq,keq)
c ****************************************************************
c       this routine calculates nonideal equilibrium constants
c       for given stage
c^Isrk and the grayson-streed modification of the chao-seader
c^Iholland, 1981, fundamentals of multicomponent distillation
c^Ipage 535
c ****************************************************************

      implicit double precision (a-z)
      integer stage,ll,jj,comp1,comp2,rrr,hhh,comp,nmc,numg
      include 'param.f'

      common/zblok/zvap(1:nmc,1:numg)
      common/consbl/gamaa,gamab,coneq(19,4)
      common/phis/tc(1:nmc),tbp(1:nmc),pc(1:nmc),
     +     w(1:nmc),vc(1:nmc)
      common/fugcons/aa0,aa1,aa2,aa3,aa4,aa5,aa6,aa7,aa8,aa9
      common/denblok/tden(1:nmc),lden(1:nmc),solpar(1:nmc)

      dimension x(1:nmc)
      dimension acalfa(1:nmc),sk(1:nmc,1:nmc),liqmolvol(1:nmc)
      dimension sb(1:nmc),alfa(1:nmc),ac(1:nmc),tr(1:nmc)
      dimension z(3),keq(1:nmc),pr(1:nmc)
c       binary interaction coefficients sk(ll,jj) are zero for
c       hydrocarbon-hydrocarbon couples
c       for others enter data
      do 8 ll=1,nmc
^Ido 9 jj=1,nmc
          sk(ll,jj)=0.0d0
 9      continue
 8    continue
      r=82.057d0
      do 7 comp=1,nmc
        tr(comp)=temp/tc(comp)
        pr(comp)=pres/pc(comp)
^If666=0.48d0+1.574d0*w(comp)-0.176d0*w(comp)**2.0d0
^If=1.0d0/tr(comp)*(1.0d0+f666*(1.0d0-tr(comp)**0.5d0))**2.0d0
^Ialfa1=(1.0d0+f666*(1.0d0-tr(comp)**0.5d0))**2.0d0
^Iac1=gamaa*r**2.0d0*tc(comp)**2.0d0/pc(comp)
^Iat=ac1*alfa1
^Icapa=at*pres/(r**2.0d0*temp**2.0d0)
^Ib=gamab*r*tc(comp)/pc(comp)
^Icapb=b*pres/(r*temp)
c       finding roots by bairstow method
 ^Icall roots(capa,capb,z)
        zvap(comp,stage)=z(3)
        liqmolvol(comp)=z(1)*r*temp/pres
 7    continue
      b=0.0d0
      tcalfa=0.0d0
      totfirst=0.0d0
      totsecond=0.0d0
      do 10 comp1=1,nmc
^Iacalfa(comp1)=0.0d0
^Itotfirst=totfirst+x(comp1)*
     +       liqmolvol(comp1)*solpar(comp1)
^Itotsecond=totsecond+x(comp1)*liqmolvol(comp1)
^Isb(comp1)=gamab*r*tc(comp1)/pc(comp1)
^Imi=.48d0+1.574d0*w(comp1)-0.176d0*w(comp1)**2.0d0
^Ialfa(comp1)=(1.0d0+mi*(1.0d0-tr(comp1)**0.5d0))**2.0d0
^Iac(comp1)=gamaa*r**2.0d0*tc(comp1)**2.0d0/pc(comp1)
^Ib=b+x(comp1)*sb(comp1)
 10   continue
      avsolpar=totfirst/totsecond
      do 11 comp1=1,nmc
```

```
^Ido 12 comp2=1,nmc
          rrr=x(comp2)*ac(comp1)**0.5d0*ac(comp2)**0.5d0*
     +         alfa(comp1)**0.5d0
          hhh=rrr*alfa(comp2)**0.5d0*(1.0d0-sk(comp1,comp2))
          acalfa(comp1)=acalfa(comp1)+hhh
          tcalfa=tcalfa+x(comp1)*hhh
 12     continue
 11   continue
      capa=tcalfa*pres/(r**2.0d0*temp**2.0d0)
      capb=b*pres/(temp*r)
      call roots(capa,capb,z)
      do 13 comp=1,nmc
^Iaaa=sb(comp)/b*(z(3)-1.0d0)-dlog(z(3)-capb)
^Ibbb=capa/capb*(2.0d0*acalfa(comp)/tcalfa-sb(comp)/b)
^Ifivap=dexp(aaa-bbb*dlog(1.0+capb/z(3)))
^Itrin=tr(comp)
^Ifi0dlog=aa0+aa1/trin+aa2*trin+aa3*trin**2.0d0+
     +       aa4*trin**3.0d0+(aa5+aa6*trin+aa7*trin**2.0d0)*pr(comp)+
     +       (aa8+aa9*trin)*pr(comp)**2.0d0-dlog10(pr(comp))
^Iif(trin.gt.1.0d0) then
          trin=1.0d0
^Iend if
        fi1dlog=-4.23893d0+8.65808d0*trin-1.22060d0/trin-
     +       3.15224d0*trin**3.0d0-0.025d0*(pr(comp)-0.6d0)
^Ifiliq=10.0d0**(fi0dlog+w(comp)*fi1dlog)
^Iactliq=dexp((liqmolvol(comp)*(solpar(comp)-avsolpar)
     +       **2.0d0)/(r*temp))
^Ikeq(comp)=actliq*filiq/fivap
 13   continue
      return
      end^I


      subroutine holdup(l,dp,voidfrac,bl,bg)
c ******************************************************************
c^Ithis routine calculates liquid and vapour volumetric holdups
c       in packed section
c       equation requires l [kg/m2 s] -
c       dp [mm] - equivalent diameter of packing
c       liq [m3 liquid/m3 of column]
c ******************************************************************

      implicit double precision (a-z)
      integer numg,nmc,ii
      include 'param.f'
      dimension bl(1:numg),bg(1:numg),l(1:numg)

      common/mixmolw/lmixwme(1:numg),vmixwme(1:numg),wme(1:nmc)
      common/liqdenbl/liqden(1:nmc,1:numg),mlden(1:numg)
      common/gasdenbl/gasden(1:nmc,1:numg),mgden(1:numg)
      do 6 ii=1,numg
        liq=0.143d0*(l(ii)/(dp*1.0d3*3600.0d0))**0.6d0
        gas=voidfrac-liq
  ^Ibl(ii)=liq*mlden(ii)/lmixwme(ii)
^Ibg(ii)=gas*mgden(ii)/vmixwme(ii)
 6    continue
      return
      end


      subroutine sgasmastran(at,dp,ky)
c ******************************************************************
c^Iindividual components gas mass transfer coefficients
c^Iby onda,et.al.,(1968) for raschig rings, berl saddles
c^Iky' [kmol/m2 hr] , g [kg/m2 hr], gasdif [m2/hr]
c ******************************************************************

      implicit double precision (a-z)
      integer comp,j,numg,nmc
      include 'param.f'
      common/diffblk/gasdif(1:nmc,1:numg),mldif(1:nmc,1:numg)
      common/gasvisbl/visgas(1:nmc,1:numg),mgvis(1:numg)
      common/gasdenbl/gasden(1:nmc,1:numg),mgden(1:numg)
      common/tempblk/ttk(1:numg)
      common/presblok/prestop,prebot,p(1:numg)
      common/molfrac/xmol(1:nmc,1:numg),ymol(1:nmc,1:numg)
      common/massflow/lr(1:numg),gasmasrat(1:numg)
      common/parastudy/alfaky,alfakx,beta1,beta2

      dimension ky(1:nmc,1:numg)
      if(dp.ge.12.0d-3) then
        c1=5.23d0
      else
        c1=2.00d0
      end if
      do 50 j=1,numg
        do 43 comp=1,nmc
          reynold=(gasmasrat(j)/(at*mgvis(j)))**0.7d0
          ky(comp,j)=c1*reynold*
     +          (visgas(comp,j)/(gasden(comp,j)*gasdif(comp,j)))**
     +          (1.0d0/3.0d0)*(at*dp)**(-2.0d0)*at*gasdif(comp,j)/
     +          (0.08206d0*ttk(j))*p(j)
          ky(comp,j)=alfaky*ky(comp,j)
 43     continue
 50   continue
      return
      end


      subroutine sliqmastran(at,aw,dp,kx)
c ******************************************************************
c^Iindividual components liquid mass transfer coefficients
c^Iby onda,et.al.,(1968) for raschig rings, berl saddles
c^Ikx' [kmol/m2 hr] , l [kg/m2 hr], liqdif [m2/hr]
c ******************************************************************

      implicit double precision (a-z)
      integer comp,j,numg,nmc
      include 'param.f'

      common/diffblk/gasdif(1:nmc,1:numg),liqdif(1:nmc,1:numg)
      common/liqvisbl/visliq(1:nmc,1:numg),mlvis(1:numg)
      common/liqdenbl/liqden(1:nmc,1:numg),mlden(1:numg)
      common/mixmolw/lmixwme(1:numg),vmixwme(1:numg),wme(1:nmc)
      common/massflow/liqmasrat(1:numg),gr(1:numg)
      common/molfrac/x(1:nmc,1:numg),y(1:nmc,1:numg)
      common/parastudy/alfaky,alfakx,beta1,beta2

      dimension kx(1:nmc,1:numg),aw(1:numg)

      do 60 j=1,numg
        do 43 comp=1,nmc
          pacnu=(liqmasrat(j)/(aw(j)*mlvis(j)))**(2.0d0/3.0d0)
          scnu=(visliq(comp,j)/(liqden(comp,j)*liqdif(comp,j)))
          kx(comp,j)=0.0051d0*pacnu
     +          *(scnu)**(-0.5d0)*(at*dp)**(0.4d0)/((liqden(comp,j)/
     +          (visliq(comp,j)/3600.0d0*9.81d0))**(1.0d0/3.0d0))*
     +          x(comp,j)*mlden(j)/lmixwme(j)*3600.0d0
          kx(comp,j)=alfakx*kx(comp,j)
 43     continue
 60   continue
      return
      end


      subroutine subdensity(j,temp,pres,density)
c ******************************************************************
c       liquid density calculations by yen and woods
c^Isherwood, 3rd ed., page 61. ^I
c ******************************************************************

      implicit double precision (a-z)
      integer jj,ii,j,comp,compon,nmc,numg
      include 'param.f'
```

```
      common/mixmolw/lmixwme(1:numg),vmixwme(1:numg),wme(1:nmc)
      common/denblok/tden(1:nmc),lden(1:nmc),solpar(1:nmc)
      common/redbl/tr(1:nmc,1:numg),tbr(1:nmc,1:numg),
     +     pr(1:nmc,1:numg)
      common/phis/tc(1:nmc),tbp(1:nmc),pc(1:nmc),
     +     w(1:nmc),vc(1:nmc)
      common/conblok/bcon,ccon,chtr(1:nmc),compon
      common/heattwo/xfeed(1:nmc)
      common/consbl/gamaa,gamab,coneq68(19,4)
      dimension kden(4)
      dimension pvp(1:nmc)
      dimension density(1:nmc)
c       saturated molar liquid density
      do 43 comp=1,nmc
^Itrnew=0.0d0
^Icall vappres(j,comp,trnew,pvp)
^Idensity(comp)=0.0d0
^Izc=pc(comp)*vc(comp)/(82.057d0*tc(comp))
^Ikden(1)=17.4425d0-214.578d0*zc+989.625d0*zc**2.0d0-
     +       1522.06d0*zc**3.0d0
^Iif (zc.le.0.26d0) then
          kden(2)=-3.28257d0+13.6377d0*zc+107.4844d0*zc**2.0d0-
     +          384.211d0*zc**3.0d0
^Ielse
          kden(2)=60.2091d0-402.063d0*zc+501.0d0*zc**2.0d0+
     +          641.0d0*zc**3.0d0
^Iend if
^Ikden(3)=0.0d0
^Ikden(4)=0.93d0-kden(2)
^Itopk=0.0d0
        do 15 ii=1,4
          topk=topk+kden(ii)*(1.0d0-tr(comp,j))**(ii/3.0d0)
 15     continue
c^Igunn and yamada for saturated liquid densities
c^Isherwood, page 60.
^Itrr=tden(comp)+273.15d0
^Ivrr=1.0d0/lden(comp)
^Iif(tr(comp,j).ge.0.2d0.and.tr(comp,j).le.0.8d0) then
          vrpart=0.33593d0-0.33953d0*tr(comp,j)+1.51941d0*
     +          tr(comp,j)**2.0d0
          vr=vrpart-2.02512d0*tr(comp,j)**3.0d0+1.11422d0*
     +          tr(comp,j)**4.0d0
^Ielse if(tr(comp,j).gt.0.8d0.and.tr(comp,j).lt.1.0d0) then
          vrpart=1.0d0+1.3d0*(1.0d0-tr(comp,j))**0.5d0*
     +          dlog10(1.0d0-tr(comp,j))
          vr=vrpart-0.5087901d0*(1.0d0-tr(comp,j))-0.91534d0*
     +          (1.0d0-tr(comp,j))**2.0d0
^Iend if
^Igama=0.29607d0-9.044999d-02*tr(comp,j)-0.04842d0*
     +       tr(comp,j)**2.0d0
^Itr1=trr/tc(comp)
        if(tr1.ge.0.2d0.and.tr1.le.0.8d0) then
          vrpart=0.33593d0-0.33953d0*tr1+1.51941d0*tr1**2.0d0
          vr1=vrpart-2.02512d0*tr1**3.0d0+1.11422d0*tr1**4.0d0
^Ielse if(tr1.gt.0.8d0.and.tr1.lt.1.0d0) then
          vrpart=1.0d0+1.3d0*(1.0d0-tr1)**0.5d0*dlog10(1.0d0-tr1)
          vr1=vrpart-0.5087901d0*(1.0d0-tr1)-0.91534d0*
     +          (1.0d0-tr1)**2.0d0
^Iend if
^Igama1=0.29607d0-9.044999d-02*tr(comp,j)-0.04842d0*
     +       tr(comp,j)**2.0d0
^Ivgunn=vrr*vr*(1.0d0-wme(comp)*gama)/(vr1*(1.0d0-wme(comp)*
     +       gama1))
^Idensity(comp)=1.0d0/vgunn
c       subcooled liquid density
^Iif (pres.lt.pvp(comp)) go to 130
^Idelpr=(pres-pvp(comp))/pc(comp)
^Itrdif=1.0d0-tr(comp,j)
^Ie1=0.714d0-(1.626d0*trdif**(1.0d0/3.0d0))-(0.646d0*trdif**
     +       (2.0d0/3.0d0))
        e2=(3.699d0*trdif)-(2.198d0*trdif**(4.0d0/3.0d0))
^Ie=e1+e2
^If1=(0.268d0*tr(comp,j)**2.0967d0)
        f2=(1.0d0+0.8d0*(-1.0d0*dlog(tr(comp,j)))**0.441d0)
^If=f1+f2
^Ig=0.05d0+4.221d0*((1.01d0-tr(comp,j))**0.75d0)*dexp(-7.848d0*
     +       (1.01d0-tr(comp,j)))
^Ih1=-10.6d0+(45.22d0*trdif**(1.0d0/3.0d0))-(103.79d0*
     +       trdif**(2.0d0/3.0d0))
        h2=(114.44d0*trdif)-(47.38d0*trdif**(4.0d0/3.0d0))
^Ih=h1+h2
^Iif ((delpr).lt.0.2d0) then
          delden=e+f*dlog(0.2d0)+g*dexp(h*0.2d0)
          deldenr=delden*delpr/0.2d0
^Ielse
          deldenr=e+f*dlog(delpr)+g*dexp(h*delpr)
^Iend if
^Iif (zc.eq.0.27d0) then
          delzc=0.0d0
          goto 130
^Iend if
c       calculation of constants in equations 68
^Iif (zc.eq.0.29d0) then
          jj=1
^Ielse if(zc.eq.0.25d0) then
          jj=2
^Ielse if(zc.eq.0.23d0) then
          jj=3
^Ielse
          jj=4
^Iend if
^Iif(jj.eq.4) then
^I  call calconst(zc,coneq68)
^Iend if^I
^Iirel=coneq68(1,jj)+coneq68(2,jj)*trdif**(1.0d0/3.0d0)+
     +       coneq68(3,jj)*trdif**(2.0d0/3.0d0)+coneq68(4,jj)*
     +       trdif+coneq68(5,jj)*trdif**(4.0d0/3.0d0)
^Ijrel=coneq68(6,jj)+coneq68(7,jj)*trdif**(1.0d0/3.0d0)+
     +       coneq68(8,jj)*trdif**(2.0d0/3.0d0)+coneq68(9,jj)*
     +       trdif+coneq68(10,jj)*trdif**(4.0d0/3.0d0)
^Ikkk=coneq68(11,jj)+coneq68(12,jj)*tr(comp,j)+
     +       coneq68(13,jj)*
     +       tr(comp,j)**2.0d0+coneq68(14,jj)*tr(comp,j)**3.0d0
^Ilrel=coneq68(15,jj)+coneq68(16,jj)*trdif**(1.0d0/3.0d0)+
     +       coneq68(17,jj)*trdif**(2.0d0/3.0d0)+coneq68(18,jj)*
     +       trdif+coneq68(18,jj)*trdif**(4.0d0/3.0d0)
^Iif (delpr.lt.0.2d0) then
^I  delzc=irel+jrel*dlog(0.2d0)+kkk*dexp(lrel*0.2d0)
^I  delzc=delzc*delpr/0.2
^Ielse
^I  delzc=irel+jrel*dlog(delpr)+kkk*dexp(lrel*delpr)
^Iend if
 130^Idensity(comp)=(1.0d0/vgunn)*((deldenr+delzc)/topk+1.0d0)
 43   continue
      return
      end



      subroutine vappres(j,comp,trnew,pvp)
c ********************************************************************
c       vapour pressure calculations by frost-kalkwarf-thodos
c^Isherwood, page 188.
c ********************************************************************

      implicit double precision (a-z)
      integer comp,j,compon,nmc,numg
      include 'param.f'

      common/redbl/tr(1:nmc,1:numg),tbr(1:nmc,1:numg),
     +     pr(1:nmc,1:numg)
      common/phis/tc(1:nmc),tbp(1:nmc),pc(1:nmc),
     +     w(1:nmc),vc(1:nmc)
      common/conblok/bcon,ccon,chtr(1:nmc),compon
```

```
      dimension pvp(1:nmc)
      external fonk
      if (trnew.eq.0.6d0) then
        chtr(comp)=0.6d0
      else
        chtr(comp)=tr(comp,j)
      end if
      bcon=(dlog(pc(comp))+2.67d0*dlog(tbr(comp,j))+27.0d0/
     +     64.0d0*((1.0d0/(pc(comp)*tbr(comp,j)**2.0d0))-1.0d0))/
     +     (1.0d0-1.0d0/tbr(comp,j)-0.7816d0*dlog(tbr(comp,j)))
      ccon=0.7816d0*bcon+2.67d0
      a10=1.0d-5
      b10=1.0d0
      eps=1.0d-10
      compon=comp
c       calculation of pvpr which is the root of fonk
      fonk1=fonk(a10)
      fonk2=fonk(b10)
      fl=min(fonk1,fonk2)
      fh=max(fonk1,fonk2)
      if(fl.eq.fonk1) then
        pl=a10
        ph=b10
      else
        pl=b10
        ph=a10
      end if
 65   psafe=0.5d0*(pl+ph)
      fonksafe=fonk(psafe)
      if(dabs(fonksafe).le.eps) then
        pvpr=psafe
        goto 75
      end if
      if(fl*fonksafe.lt.0.0d0) then
        ph=psafe
        fh=fonksafe
      else
        if(fonksafe*ph.lt.0.0d0) then
          pl=psafe
          fl=fonksafe
        end if
      end if
      goto 65
 75   pvp(comp)=pvpr*pc(comp)
      return
      end



      subroutine sgasdensity(j,temp,pres,gasden)
c ***********************************************************************
c     it calculates the gas density of a component using ideal gas law
c ***********************************************************************

      implicit double precision (a-z)
      integer comp,j,nmc,numg
      include 'param.f'

      common/mixmolw/lmixwme(1:numg),vmixwme(1:numg),wme(1:nmc)
      common/zblok/zvap(1:nmc,1:numg)

      dimension gasden(1:nmc)
      do 43 comp=1,nmc
        gasden(comp)=pres*wme(comp)/(zvap(comp,j)*8.205d-2*temp)
 43   continue
      return
      end



      subroutine sliqviscos(temp,lviscos)
c ***********************************************************************
c     it calculates the liquid viscosity of a component, [kg /m hr]
c     using the method given by Reid et.al. 3th ed.(1977), page 629
c ***********************************************************************

      implicit double precision (a-z)
      integer comp,nmc,numg
      include 'param.f'
      common/visbl/visa(1:nmc),visb(1:nmc)
      dimension lviscos(1:nmc)
      do 43 comp=1,nmc
        lviscos(comp)=10.0d0**(visa(comp)*((1.0d0/temp)-
     +       (1.0d0/visb(comp))))*0.001d0*3600.0d0
 43   continue
      return
      end



      subroutine mixsgasviscos(temp,y,mixgvis)
c ***********************************************************************
c^Idean and stiel for viscosity of gas mixtures
c^Isherwood, page 419. mgvis [kg/m hr]
c ***********************************************************************

      implicit double precision (a-z)
      integer comp,nmc,numg
      include 'param.f'

      common/phis/tc(1:nmc),tbp(1:nmc),pc(1:nmc),w(1:nmc),
     +     vc(1:nmc)
      common/mixmolw/lmixwme(1:numg),vmixwme(1:numg),wme(1:nmc)

      dimension y(1:nmc)
c^Ikay's rule for the pseudocritical temperature
      tcm=0.0d0
      topzc=0.0d0
      topvc=0.0d0
      topw=0.0d0
      do 24 comp=1,nmc
^Itcm=tcm+y(comp)*tc(comp)
^Izc=pc(comp)*vc(comp)/(82.057d0*tc(comp))
^Itopzc=topzc+y(comp)*zc
^Itopvc=topvc+y(comp)*vc(comp)
^Itopw=topw+y(comp)*wme(comp)
 24   continue
      pcm=82.057d0*topzc*tcm/topvc
      trm=temp/tcm
      epsm=(tcm**(1.0d0/6.0d0))/(pcm**(2.0d0/3.0d0)*topw**
     +     (1.0d0/2.0d0))
      if(trm.le.1.5d0) then
        mixgvis=(3.4d0*trm**(8.0d0/9.0d0))/epsm
      else
        mixgvis=(16.68d0*(0.1338d0*trm-0.0932d0)**
     +       (5.0d0/9.0d0))/epsm
      end if
      mixgvis=mixgvis*1d-7*3600.0d0
 76   continue
      return
      end



      subroutine sgasviscos(temp,gviscos)
c ***********************************************************************
c^Igas viscosity by chapman and enskog theory
c^Ireid, et.al. page 394-396, gasvis [kg/m hr]
c ***********************************************************************

      implicit double precision (a-z)
      integer comp,nmc,numg
      include 'param.f'

      common/phis/tc(1:nmc),tbp(1:nmc),pc(1:nmc),w(1:nmc),vc(1:nmc)
      common/mixmolw/lmixwme(1:numg),vmixwme(1:numg),wme(1:nmc)
```

```
      dimension gviscos(1:nmc)
      a=1.16145d0
      b=0.14874d0
      c=0.52487d0
      d=0.7732d0
      e=2.16178d0
      f=2.43787d0
      do 43 comp=1,nmc
^Ityildiz=temp/((0.7915d0+0.1693d0*w(comp))*tc(comp))
^Iif(tyildiz.ge.0.3d0.and.tyildiz.le.100.0d0) then
          ohm=(a/tyildiz**b)+(c/dexp(d*tyildiz))+(e/dexp(f*tyildiz))
          sigma=(2.3551d0-0.087d0*w(comp))/(pc(comp)/tc(comp))**
     +         (1.0d0/3.0d0)
          gviscos(comp)=(26.69d0*(wme(comp)*temp)**0.5d0/
     +         (sigma**2.0d0*ohm))*1d-7*3600.0d0
        else
        end if
 43   continue
      return
      end


      subroutine sliqsurten(j,temp,x,lsurface,mixliqst)
c ********************************************************************
c^Iliquid surface tension by hakim et al. (reid, p 608)
c^Iunit [n/m]
c ********************************************************************

      implicit double precision (a-z)
      integer j,comp,nmc,numg
      include 'param.f'
      common/phis/tc(1:nmc),tbp(1:nmc),pc(1:nmc),
     +     w(1:nmc),vc(1:nmc)
      common/redbl/tr(1:nmc,1:numg),tbr(1:nmc,1:numg),
     +     pr(1:nmc,1:numg)

      dimension lsurface(1:nmc)
      dimension x(1:nmc),pvp(1:nmc)
      trnew=0.6d0
      mixliqst=0.0d0
      do 21 comp=1,nmc
^Icall vappres(j,comp,trnew,pvp)
^Ixfac=dlog10(pvp(comp)/pc(comp))+1.7d0*w(comp)+1.552d0
^Im=1.21d0+0.5385d0*w(comp)-14.61d0*xfac-32.07d0*xfac**
     +       2.0d0-1.656d0*w(comp)*7d0*xfac**
     +       2.0d0-1.656d0*w(comp)**2.0d0+22.03d0*w(comp)*xfac
^Iqp=0.1574d0+0.359d0*w(comp)-1.769d0*xfac-13.69d0*
     +       xfac**2.0d0-0.51d0*w(comp)**2.0d0+1.298d0*w(comp)*xfac
^Ilsurface(comp)=(pc(comp)**(2.0d0/3.0d0)*tc(comp)**
     +       (1.0d0/3.0d0)*qp*((1.0d0-tr(comp,j))/0.4d0)**m)*1.0d-3
^Imixliqst=mixliqst+x(comp)*lsurface(comp)**(-2.0d0)
 21   continue
      mixliqst=dexp(dlog(mixliqst)/(-2.0d0))
      return
      end


      subroutine bingasdif(temp,pres,compa,compb,binnum,bingdif)
c ********************************************************************
c^Ibinary gas diffusivity, [m2/h]
c^Iwilke-lee modification to chapman-enskog relationship (1955)
c^Isource : reid, et.al., 'the properties of gases and liquids', p.555
c ********************************************************************

      implicit double precision (a-z)
      integer ii1,i1,compa,compb,binnum,nmc,numg
      include 'param.f'

      common/mixmolw/lmixwme(1:numg),vmixwme(1:numg),wme(1:nmc)
      common/phis/tc(1:nmc),tbp(1:nmc),pc(1:nmc),w(1:nmc),vc(1:nmc)

      dimension v(1:nmc),epkey(2),sigma(2)

c^Inumber of binaries is (ncomp!/2! (ncomp-2)!)
c^Ireenter the bingdif dimension
      dimension bingdif(1:45)
      a=1.06036d0
      b=0.15610d0
      c=0.19300d0
      d=0.47635d0
      e=1.03587d0
      f=1.52996d0
      g=1.76474d0
      h=3.89411d0
      do 55 ii1=1,2
^Iif (ii1.eq.1) then
          i1=compa
        else
          i1=compb
^Iend if
^Iepkey(ii1)=1.15d0*(tbp(i1)+273.15d0)
c^Iv is calculated by tyn and calus method page 59, reid et al.
^Iv(i1)=0.285d0*vc(i1)**1.048d0
^Isigma(ii1)=1.18d0*v(i1)**(1.0d0/3.0d0)
        epabk=(epkey(1)*epkey(2))**.5d0
 55   continue
      tstar=temp/epabk
      sigmaab=(sigma(1)+sigma(2))/2.0d0
      gamad=(a/tstar**b)+(c/dexp(d*tstar))+(e/dexp(f*tstar))+
     +     (g/dexp(h*tstar))
      ratio=((wme(compa)+wme(compb))/(wme(compa)*wme(compb)))**.5d0
c^Ibinary gas diffusivity for binnum th binary [m2/hr]
      bingdif(binnum)=((0.00217d0-0.0005d0*ratio)*temp**
     +     (3.0d0/2.0d0)*ratio/(pres*sigmaab**2.0*gamad))*
     +     1.0d-4*3600.0d0
c^I-----------------------------------------------------
c^Isecond method for binary gas diffusivity
c^Ithe wilke lee modification of the hirschfelder-bird-spotz method
c^Itreybal, 'mass transfer operations', p.31
c
      return
      end



      subroutine binliqdif(j,temp,pres,compa,compb,x,
     +     binnum,binldif)
c ********************************************************************
c^Ibinary liquid diffusivity [m2/hr]
c^Ischeibel correlation
c^Ireid et. al., page 571 - perry, page 3-287
c^Ia is solute
c^Ib is solvent
c^Iv is the molal liquid volume [cm3/mol] calculated by tyn and
c^Icalus method (reid, page 59)
c       convert liqvis and mlvis values from [kg/m h] to [kg/m s]
c ********************************************************************

      implicit double precision (a-z)
      integer compa,compb,numg,nmc,binnum,j
      include 'param.f'

      common/name/namecomp
      common/mixmolw/lmixwme(1:numg),vmixwme(1:numg),wme(1:nmc)
      common/phis/tc(1:nmc),tbp(1:nmc),pc(1:nmc),w(1:nmc),vc(1:nmc)
      common/liqvisbl/liqvis(1:nmc,1:numg),mlvis(1:numg)

      dimension vol(1:nmc)
      dimension binldif(1:45)
      dimension xass1(1:nmc)
      dimension x(1:nmc)
      dimension xass2(1:nmc),keq(1:nmc)
      character *15 namecomp(1:nmc)
```

```
      xass1(compa)=0.0d0
      xass1(compb)=0.0d0
      xass2(compa)=0.0d0
      xass2(compb)=0.0d0

      vol(1)=.285d0*vc(compa)**1.048d0
      vol(2)=.285d0*vc(compb)**1.048d0
c^Idiffusivity of a in very dilute solution in solvent b
c^Iif b=water and v(1) < v(2) then kcal=25.2d-08
c^Iif b=benzene and v(1) < 2v(2) then kcal=18.9d-08
c^Ifor other solvents if v(1) < 2.5v(2) then kcal=17.5d-08
c
      if (namecomp(compb).eq.'water' .and. vol(1).lt.vol(2)) then
        kcal=25.2d-08
      else if(namecomp(compb).eq.'benzene'
     +       .and.vol(1).lt.(2.0d0*vol(2))) then
        kcal=18.9d-08
      else if (vol(1) .lt. (2.5d0*vol(2))) then
        kcal=17.5d-08
      else
        kcal=8.2d-08*(1.0d0+(3.0d0*vol(2)/vol(1))**(2.0d0/3.0d0))
      end if

      dab0=kcal*temp/(liqvis(compb,j)/3600.0d0*1.0d3*vol(1)**
     +     (1.0d0/3.0d0))
      if (namecomp(compa).eq.'water' .and. vol(2).lt.vol(1)) then
        kcal=25.2d-08
      else if(namecomp(compa).eq.'benzene'
     +       .and.vol(2).lt.(2.0d0*vol(1))) then
        kcal=18.9d-08
      else if (vol(2) .lt. (2.5d0*vol(1))) then
        kcal=17.5d-08
      else
        kcal=8.2d-08*(1.0d0+(3.0d0*vol(1)/vol(2))**(2.0d0/3.0d0))
      end if
      dba0=kcal*temp/(liqvis(compa,j)/3600.0d0*1.0d3*vol(2)**
     +     (1.0d0/3.0d0))
c^Ithe diffusivity in concentrated solutions
      xass1(compa)=0.5d0
      xass1(compb)=0.5d0
      call kval(1,temp,pres,xass1,act1,keq)
c        act3=act1
      xass2(compa)=0.3d0
      xass2(compb)=0.7d0
      call kval(1,temp,pres,xass2,act2,keq)

      diff1=(dlog10(act1)-dlog10(act2))/
     +     (dlog10(xass1(compa))-dlog10(xass2(compa)))

c^Ixass2(compa)=0.7d0
c^Ixass2(compb)=0.3d0
c^Icall kval(1,temp,pres,xass2,act4,keq)
c^Idiff2=(dlog10(act3)-dlog10(act4))/
c     +  (dlog10(xass1(compb))-dlog10(xass2(compb)))

c     +  liqvis(compa,j),liqvis(compb,j),x(compa),
c     +  x(compb),dab0,diff1,mlvis(j)

      binldif(binnum)=((dba0*liqvis(compa,j)/3600.0d0*1.0d3)**
     +     x(compa))*((dab0*liqvis(compb,j)/3600.0d0*1.0d3)**
     +     x(compb))*(1.0d0+diff1)/(mlvis(j)/3600.0d0*1.0d3)*
     +     1.0d-4*3600.0d0

c^Iabinldif(j)=((dba0*liqvis(compa,j)*1e3)**x(compa))*
c     +  ((dab0*liqvis(compb,j)*1.0d3)**x(compb))*(1.0d0+diff2)/
c     +  (mlvis(j)*1.0d3)*1.0d-4

      return
      end


      subroutine mpromat(grup,ncomp,molfrac,pureprop,mixprop)
c ******************************************************************
c^Imixture property calculations
c ******************************************************************

      implicit double precision (a-z)
      integer grup,stage,ncomp,comp
      dimension molfrac(1:ncomp,1:grup)
      dimension mixprop(1:grup),pureprop(1:ncomp,1:grup)
      do 78 stage=1,grup
^Itopmix=0.0d0
^Ido 30 comp=1,ncomp
          topmix=topmix+pureprop(comp,stage)*molfrac(comp,stage)
 30     continue
^Imixprop(stage)=topmix
 78   continue
      return
      end


      subroutine smgdiffus(temp,pres,y,mixgasdif)
c ******************************************************************
c^Ieffective diffusion coefficient in a multicomponent mixture
c^Iwilke c.r., 'diffusional properties of multicomponent gases'
c^Ichem.eng.prog., 46(2), (1950) p.95
c ******************************************************************

      implicit double precision (a-z)
      integer nmc,binnum,compa,compb,numg
      include 'param.f'
      common/mixmolw/lmixwme(1:numg),vmixwme(1:numg),wme(1:nmc)
      common/phis/tc(1:nmc),tbp(1:nmc),pc(1:nmc),w(1:nmc),vc(1:nmc)
      dimension y(1:nmc),bingdif(1:45)
      dimension mixgasdif(1:nmc)
c^Icompa is the first component of the binary
c^Icompb is the second component of the binary
      binnum=0
      do 43 compa=1,nmc
^Itop=0.0d0
^Ido 17 compb=1,nmc
          if (compa.ne.compb) then
            binnum=binnum+1
            call bingasdif(temp,pres,compa,compb,binnum,bingdif)
            top=top+y(compb)/bingdif(binnum)
          end if
 17     continue
^Imixgasdif(compa)=(1.0d0-y(compa))/top
 43   continue
 79   continue
      return
      end


      subroutine smldiffus(j,temp,pres,x,mixliqdif)
c ******************************************************************
c^Ieffective diffusion coefficient in a multicomponent mixture
c^Iwilke c.r., 'diffusional properties of multicomponent liquids'
c^Ichem.eng.prog., 46(2), (1950) p.95
c ******************************************************************

      implicit double precision (a-z)
      integer j,nmc,binnum,compa,compb,numg
      include 'param.f'
      common/name/namecomp
      common/mixmolw/lmixwme(1:numg),vmixwme(1:numg),wme(1:nmc)
      common/phis/tc(1:nmc),tbp(1:nmc),pc(1:nmc),w(1:nmc),vc(1:nmc)
      common/liqvisbl/liqvis(1:nmc,1:numg),mlvis(1:numg)

      dimension x(1:nmc),binldif(1:45)
      dimension mixliqdif(1:nmc)
      character *15 namecomp(1:nmc)

c^Icompa is the first component of the binary
c^Icompb is the second component of the binary
      binnum=0
```

```
      do 43 compa=1,nmc
^Itop=0.0d0
^Ido 17 compb=1,nmc
          if (compa.ne.compb) then
            binnum=binnum+1
            call binliqdif(j,temp,pres,compa,compb,x,binnum,
     +           binldif)
^Itop=top+x(compb)/binldif(binnum)
      end if
 17   continue
      mixliqdif(compa)=(1.0d0-x(compa))/top
 43   continue

      return
      end

      subroutine specintarea(at,liqrate,pacsurten,intarea)
c ********************************************************************
c^Igas-liquid interficial area calculations by onda et. al.
c ********************************************************************

      implicit double precision (a-z)
      integer j,nmc,numg
      include 'param.f'
      common/surfacbl/liqsur(1:nmc,1:numg),mlst(1:numg)
      common/liqvisbl/liqvis(1:nmc,1:numg),mlvis(1:numg)
      common/liqdenbl/liqden(1:nmc,1:numg),mlden(1:numg)
      dimension liqrate(1:numg)
      dimension intarea(1:numg)
c^Itime unit is in seconds
c^Iliqrate [kg/m2 s], mlvis [kg/m s]
      grav=9.81d0
      do 80 j=1,numg
        re=0.0d0
        fr=0.0d0
        we=0.0d0
        bbb=0.0d0
        re=liqrate(j)/(at*mlvis(j))
        fr=(liqrate(j)/3600.0d0)**2.0d0*at/(mlden(j)
     +       **2.0d0*grav)
        we=(liqrate(j)/3600.0d0)**2.0d0/(mlden(j)*mlst(j)*at)
^Ibbb=-1.45d0*(pacsurten/mlst(j))**0.75d0*
     +       (re)**0.1d0*(fr)**(-0.05d0)*(we)**0.2d0
        intarea(j)=at*(1.0d0-dexp(bbb))
 80   continue
      return
      end


      subroutine cpliquid(temp,liqcp)
c ********************************************************************
c       liquid heat capacity [j/mol k] by sternling-brown
c       converted to [kj/kg k]
c       reid et.al. page 156
c ********************************************************************

      implicit double precision (a-z)
      integer nmc,comp,numg
      include 'param.f'
      common/phis/tc(1:nmc),tbp(1:nmc),pc(1:nmc),w(1:nmc),vc(1:nmc)
      common/cpblok/cpa(1:nmc),cpb(1:nmc),cpc(1:nmc),cpd(1:nmc)
      common/mixmolw/lmixwme(1:numg),vmixwme(1:numg),wme(1:nmc)
      dimension liqcp(1:nmc)
      r=82.057d0
c       r [cm3 atm / mol k]
      do 34 comp=1,nmc
        gascp=cpa(comp)+cpb(comp)*temp+cpc(comp)*
     +       temp**2+cpd(comp)*temp**3
        tr=temp/tc(comp)
        liqcp(comp)=gascp+r*((0.5d0+2.2d0*w(comp))*(3.67d0+
     +       11.64d0*(1.0d0-tr)**4.0d0)+0.634d0*(1.0d0-tr)**-1.0d0)
c       liqcp [j/mol k]
        liqcp(comp)=liqcp(comp)/wme(comp)
c       liqcp [kj/kg k]
 34   continue
      return
      end


      subroutine tconliquid(grup,liqcp,liqk)
c ********************************************************************
c       liquid thermal conductivity [kj/m k hr]
c       weber equation coulson vol.6 page 244
c       liqcp [kj/kg k]
c ********************************************************************

      implicit double precision (a-z)
      integer nmc,comp,grup,numg
      include 'param.f'

      common/mixmolw/lmixwme(1:numg),vmixwme(1:numg),wme(1:nmc)
      common/liqdenbl/liqden(1:nmc,1:numg),mlden(1:numg)
      dimension liqcp(1:nmc)
      dimension liqk(1:nmc)
      do 54 comp=1,nmc
        liqk(comp)=3.56d-5*liqcp(comp)*
     +       (liqden(comp,grup)**4.0d0/wme(comp))**(1.0d0/3.0d0)*3.6d0
 54   continue
      return
      end

      subroutine heattrmix(grup,pres,tfilm,twall,temp,
     +     lcp,heato)
      implicit double precision (a-z)
      integer grup,nmc,comp,numg
      include 'param.f'
      common/heatfirst/hod,hid,coildo,coildi,kw,heati,acoil,
     +     tempsteam
      common/heattwo/xfeed(1:nmc)

      dimension liqcp(1:nmc),liqk(1:nmc)
      dimension liqvis(1:nmc),liqden(1:nmc)

      call cpliquid(tfilm,liqcp)
      call tconliquid(1,liqcp,liqk)
      call sliqviscos(tfilm,liqvis)
      call subdensity(1,tfilm,pres,liqden)
      lcp=0.0d0
      lden=0.0d0
      lk=0.0d0
      lvis=0.0d0
      do 63 comp=1,nmc
        lcp=lcp+liqcp(comp)*xfeed(comp)
        lden=lden+liqden(comp)*xfeed(comp)
        lk=lk+liqk(comp)*xfeed(comp)
        lvis=lvis+liqvis(comp)*xfeed(comp)
 63   continue
      npr=(lcp*1.0d3*lvis)/lk
      beta=1.0d0/tfilm
      diftemp=twall-temp
      ngr=(beta*9.81d0*diftemp*coildo**3.0d0*lden**2.0d0)/
     +     lvis**2.0d0
      yval=ngr*npr
      if (yval.gt.1.0d-5.and.yval.lt.1.0d-3) then
        a=0.71d0
        m=1.0d0/25.0d0
      end if
      if (yval.gt.1.0d-3.and.yval.lt.1.0d0) then
        a=1.09d0
        m=1.0d0/10.0d0
      end if
      if (yval.gt.1.0d0.and.yval.lt.1.0d4) then
        a=1.09d0
        m=1.0d0/5.0d0
      end if
      if (yval.gt.1.0d4.and.yval.lt.1.0d9) then
```

```
        a=0.53d0
        m=1.0d0/4.0d0
      end if
      if (yval.gt.1.0d9) then
        a=0.13d0
        m=1.0d0/3.0d0
      end if
      nnus=a*yval**m
      heato=nnus*lk/coildo
      return
      end

      subroutine heatboil(grup,q,liqden,tfilm,twall,temp,pres
     +     lcp,htrboil)
      implicit double precision (a-z)
      integer grup,nmc,comp,numg
      include 'param.f'
      common/heatfirst/hod,hid,coildo,coildi,kw,heati,acoil,
     +     tempsteam
      common/heattwo/xfeed(1:nmc)
      dimension liqden(1:nmc)
      dimension liqcp(1:nmc),liqk(1:nmc)
      dimension liqvis(1:nmc),gasden(1:nmc)
      dimension liqsur(1:nmc)
      call cpliquid(tfilm,liqcp)
      call tconliquid(1,liqcp,liqk)
      call sliqviscos(tfilm,liqvis)
      call sgasdensity(1,tfilm,pres,gasden)
      lcp=0.0d0
      lden=0.0d0
      lk=0.0d0
      lvis=0.0d0
      gden=0.0d0
      do 63 comp=1,nmc
        lcp=lcp+liqcp(comp)*xfeed(comp)
        lden=lden+liqden(comp)*xfeed(comp)
        lk=lk+liqk(comp)*xfeed(comp)
        lvis=lvis+liqvis(comp)*xfeed(comp)
        gden=gden+gasden(comp)*xfeed(comp)
 63   continue
      call enthal(xfeed,xfeed,tfilm,pres,mixhl,mixhv,vv)
      lamda=mixhv-mixhl
      call sliqsurten(1,tfilm,xfeed,liqsur,mlst)
      npr=(lcp*lvis)/(lk*1.0d-3)
      htrboil=0.225d0*lk/coildo*npr**0.69d0*
     +     (q*coildo/(lamda*1.0d3*lvis))*
     +     *0.69d0*(pres*101.325d3*coildo/mlst)**0.31d0*
     +     (lden/gden-1.0d0)**0.33d0
      return
      end

      subroutine axial(crosarea,dp,laxdis,gaxdis)
c ********************************************************************
c       axial dispersion coefficients [m2/hr]
c ********************************************************************

      implicit double precision (a-z)
      integer nmc,numg,j
      include 'param.f'
      common/massflow/lrate(1:numg),grate(1:numg)
      common/liqdenbl/liqden(1:nmc,1:numg),mlden(1:numg)
      common/gasdenbl/gasden(1:nmc,1:numg),mgden(1:numg)
      common/gasvisbl/gasvis(1:nmc,1:numg),mgvis(1:numg)
      common/liqvisbl/liqvis(1:nmc,1:numg),mlvis(1:numg)
      common/parastudy/alfaky,alfakx,beta1,beta2

      dimension laxdis(1:numg)
      dimension gaxdis(1:numg)
c       conversion factor from [lbm/h ft2] to [kg/s m2]
      cv=1.35623d-3

c       conversion of lrate, grate, mlvis, mgvis from hr to sec
      do 10 j=1,numg
        liqrate=lrate(j)/3600.0d0
        gasrate=grate(j)/3600.0d0
        mixliqvis=mlvis(j)/3600.0d0
        mixgasvis=mgvis(j)/3600.0d0
        mixlden=mlden(j)
        mixgden=mgden(j)
c       ---------------------------------------------------------------
c       calculation of liquid phase axial dispertion coefficient [m2/s]
c       for metal pall rings
c       k. b. kushalkar and v. g. pangarkar
c       chem. eng. sci. 45(3), p.759-763, (1990)

        vl=liqrate/mixlden
        vd=vl*(1.0d0+0.0408d0*vl**(-0.597d0))/(0.1024d0*vl**0.527d0)
        dd=(4.0d0*vl*crosarea/(3.1416d0*vd))**0.5d0
        ped=2.195d-6*(dd*vd*mixlden/mixliqvis)**1.46d0
        laxdis(j)=vd*dd/ped*3600.0d0*beta2
c       ---------------------------------------------------------------
c       v. e. sater and o. levenspiel
c       i&ec fundementals, 5(1), p.86-92, 1966
        pel=(7.58d-3*(dp*liqrate/mixliqvis)**0.703d0)
        laxdis(j)=(dp*liqrate/mixlden)/pel*3600.0d0*beta2
c       ---------------------------------------------------------------
c       w.e. dunn, et.al., ind.eng.chem.fundm., 16(1),1977, p.116
c        if(2.71246d0.le.liqrate.and.14.91853d0.gt.liqrate.and.
c     +       0.0d0.le.gasrate.and.1.49d0.gt.gasrate) then
c       1" berl saddles
c          pel=0.033d0*10.0d0**(4.93d-5*liqrate/cv)
c       1" rasching rings
c          pel=0.038d0*10.0d0**(4.93d-5*liqrate/cv)
c       2" rasching rings
c          pel=0.051d0*10.0d0**(4.93d-5*liqrate/cv)
c        end if

c       ---------------------------------------------------------------
c       p.co and r.bibaud, the can.j. of chem.eng., 49, 1971, p.727
c        if(5.42492d0.le.liqrate.and.20.343d0.ge.liqrate) then
c          rel=dp*liqrate/mixliqvis
c          pel=dexp(-4.105d0)*rel**0.4313d0
c        end if
c       ---------------------------------------------------------------
c       ---------------------------------------------------------------
        gas phase axial dispersion
        reg=dp*gasrate/mixgasvis
        rel=dp*liqrate/mixliqvis
        peg=3.4d0*(reg)**(-0.67d0)
     +       *10.0d0**(-0.0026d0*rel)
        gaxdis(j)=(dp*gasrate/mixgden)/peg*3600.0d0
c       ---------------------------------------------------------------
c       w.e. dunn, et.al., ind.eng.chem.fundm., 16(1),1977, p.116
c        if(0.0d0.le.liqrate.and.149.1853d0.gt.liqrate.and.
c     +       0.407d0.le.gasrate.and.1.49d0.gt.gasrate) then
c       1" berl saddles
c          peg=(0.822d0-4.73d-4*gasrate/cv)*10.0d0**(-3.85d-5*liqrate/cv)
c       1" rasching rings
c          peg=(0.665d0-3.83d-4*gasrate/cv)*10.0d0**(-3.85d-5*liqrate/cv)
c       2" rasching rings
c          peg=(0.756d0-1.875d-4*gasrate/cv)*10.0d0**
c     +         (-1.61d-5*liqrate/cv)
c        gaxdis(j)=(dp*gasrate/mixgden)/peg*3600.0d0
c        end if


c        parametric study
        gaxdis(j)=gaxdis(j)*beta1

 10   continue

      return
      end


      subroutine tempfonk(x,keq,functemp)
```

```
c ***********************************************************************
c     bubble point calculations
c ***********************************************************************

      implicit double precision (a-z)
      integer nmc,ii,numg
      include 'param.f'
      dimension x(1:nmc),keq(1:nmc)
      topkx=0.0d0
      do 61 ii=1,nmc
        topkx=topkx+keq(ii)*x(ii)
 61   continue
      functemp=topkx-1.0d0
      return
      end


      subroutine tempfunctur(x,keq,step,functemp,functur)
c ***********************************************************************
c     derivative function calculation for bubble point calculations
c ***********************************************************************

      implicit double precision (a-z)
      integer nmc,ii,numg
      include 'param.f'
      dimension x(1:nmc),keq(1:nmc)
      topkx=0.0d0
      do 61 ii=1,nmc
        topkx=topkx+keq(ii)*x(ii)
 61   continue
      functur=((topkx-1.0d0)-functemp)/step
      return
      end



      subroutine calconst(zc,coneq68)
c ***********************************************************************
c       interpolation to find discrete values of constants
c ***********************************************************************

      implicit double precision (a-z)
      integer first,second,ii
      dimension coneq68(19,4)
      if (zc.lt.0.29d0.and.zc.gt.0.25d0) then
        first=1
        second=2
        zdif=zc-0.25d0
        zdiv=0.29d0-0.25d0
      else if(zc.lt.0.25d0.and.zc.gt.0.23d0) then
        first=2
        second=3
        zdif=zc-0.23d0
        zdiv=0.25d0-0.23d0
      end if
      do 16 ii=1,19
^Iconeq68(ii,4)=(coneq68(ii,first)-coneq68(ii,second))/zdiv*
     +       zdif+coneq68(ii,second)
 16   continue
      return
      end

      double precision function fonk(pvpr)
      implicit double precision (a-z)
      integer compon,nmc,numg
      include 'param.f'
      common/conblok/bcon,ccon,chtr(1:nmc),compon
      fonk=(dlog(pvpr)-bcon*(1.0d0/chtr(compon)-1.0d0)-ccon*
     +     dlog(chtr(compon))-27.0d0/64.0d0*(pvpr/chtr(compon))**
     +     2.0d0-1.0d0))
      return
      end


      subroutine gauss(n,grup,z,b,back)
c ***********************************************************************
c     gauss elimination method for matrix decomposition
c       2*numg is the number of variables that should be
c       calculated in packed section
c       z must contain the elements of the real matrix
c       b must contain the elements of right hand side
c       n is the order of matrix
c ***********************************************************************

      implicit double precision (a-z)
      integer grup,n,nmc,numg,m,l,i,j,jj,k,kp1,nn,ip1
      include 'param.f'
      dimension z(n,n),b(n)
      dimension a(numg*2,numg*2+1),back(numg*2)

      m=n+1
      l=n-1
c       a is the augmented matrix
      do 23 i=1,n
        do 24 j=1,n
          a(i,j)=z(i,j)
 24     continue
        a(i,m)=b(i)
 23   continue

c       solution of simultaneous equations by gauss elimination
      do 12 k=1,l
        jj=k
        big=dabs(a(k,k))
        kp1=k+1
        do 7 i=kp1,n
          ab=dabs(a(i,k))
          if(big-ab) 6,7,7
 6        big=ab
          jj=i
 7      continue
        if(jj-k)8,10,8
 8      do 9 j=k,m
          temp=a(jj,j)
          a(jj,j)=a(k,j)
          a(k,j)=temp
 9      continue
 10     do 11 i=kp1,n
          quot=a(i,k)/a(k,k)
          do 111 j=kp1,m
            a(i,j)=a(i,j)-quot*a(k,j)
 111      continue
 11     continue
        do 112 i=kp1,n
          a(i,k)=0.0d0
 112    continue
 12   continue
      back(n)=a(n,m)/a(n,n)
      do 14 nn=1,l
        sum=0.0d0
        i=n-nn
        ip1=i+1
        do 13 j=ip1,n
          sum=sum+a(i,j)*back(j)
          back(i)=(a(i,m)-sum)/a(i,i)
 13     continue
 14   continue

      return
      end  ^I


      subroutine roots(capa,capb,x88)
c ***********************************************************************
c       **** roots of polynomial of z by bairstow method ***
c ***********************************************************************
```

```
4235
4236        implicit double precision (a-z)
4237        integer i8,kok,n55,n5,n56
4238        dimension ac(6),kat(6),bc(6),dc(6),x88(3)
4239        x88(1)=0.0d0
4240        x88(2)=0.0d0
4241        x88(3)=0.0d0
4242        n5=3
4243        n55=3
4244        kok=0
4245        ac(3)=-capa*capb
4246        kat(1)=ac(3)
4247        ac(4)=capa-capb-capb**2.0d0
4248        kat(2)=ac(4)
4249        ac(5)=-1.0d0
4250        kat(3)=ac(5)
4251        ac(6)=1.0d0
4252        kat(4)=ac(6)
4253        p5=1.0d0
4254        q5=1.0d0
4255   180  bc(n5+1)=ac(n5+n5)
4256        bc(n5)=ac(n5+n5-1)-p5*bc(n5+1)
4257        do 220 i8=n5-1,1,-1
4258          bc(i8)=ac(i8+n5-1)-p5*bc(i8+1)-q5*bc(i8+2)
4259   220  continue
4260        dc(n5+1)=0.0d0
4261        dc(n5+2)=0.0d0
4262        do 260 i8=n5,2,-1
4263          dc(i8)=-bc(i8+1)-p5*dc(i8+1)-q5*dc(i8+2)
4264   260  continue
4265        d45=dc(2)**2.0d0+p5*dc(3)*dc(2)+q5*dc(3)**2.0d0
4266        p666=bc(2)*(dc(2)+p5*dc(3))-(bc(1)+p5*bc(2))*dc(3)
4267        p6=p5-p666/d45
4268        q6=q5-((bc(1)+p5*bc(2))*dc(2)+dc(3)*bc(2)*q5)/d45
4269        top=((p6-p5)**2+(q6-q5)**2)**.5d0
4270        if(top.le.1.0d-10)goto 340
4271        p5=p6
4272        q5=q6
4273        goto 180
4274   340  dis=p6**2.0d0-4.0d0*q6
4275        if(dis.lt.0.0d0)goto 440
4276        x88(kok+1)=(-p6+dis**.5d0)/2.0d0
4277        x88(kok+2)=(-p6-dis**.5d0)/2.0d0
4278        kok=kok+2
4279   440  if((n5-2).gt.1)goto 520
4280        x88(kok+1)=-bc(n5)/bc(n5+1)
4281        kok=kok+1
4282        goto 900
4283   520  n56=n5
4284        n5=n5-2
4285        do 560 i8=1,n5+1
4286          ac(n5+n5-i8+1)=bc(n56+1-i8+1)
4287   560  continue
4288        if(n5.eq.2)then
4289          goto 910
4290        else
4291          goto 580
4292        end if
4293   580  goto 900
4294   910  dis=ac(n5+1)**2.0d0-4.0d0*ac(n5+2)*ac(n5)
4295        if(dis.lt.0.0d0)goto 960
4296        x88(kok+1)=(-ac(n5+1)+sqrt(dis))/(2.0d0*ac(n5+2))
4297        x88(kok+2)=(-ac(n5+1)+sqrt(dis))/(2.0d0*ac(n5+2))
4298        kok=kok+2
4299   960  goto 580
4300  c       ordering of roots from small to big
4301   900  z1=min(x88(1),x88(2),x88(3))
4302        z3=max(x88(1),x88(2),x88(3))
4303        x88(1)=z1
4304        x88(3)=z3
4305        return
4306        end
4307
```

```
4308
4309        subroutine dres1(t,sol,yprime,delta,ires,rpar,ipar)
4310  c ***********************************************************************
4311  c     dynamic mass transfer equations written in  dassl code
4312  c ***********************************************************************
4313
4314        implicit double precision (a-z)
4315        integer numg,nmc,j,comp,ires,ipar,kil
4316        integer yaztot,yazpro,option,decide,model,opmode,simpro
4317        include 'param.f'
4318        common/molfrac/xmolfrac(1:nmc,1:numg),ymolfrac(1:nmc,1:numg)
4319        common/disperblok/laxdis(1:numg),gaxdis(1:numg)
4320        common/mastran/kx(1:nmc,1:numg),ky(1:nmc,1:numg)
4321        common/propla/a(1:numg),keqcons(1:nmc,1:numg)
4322        common/premolfrac/xpre(1:nmc,1:numg),ypre(1:nmc,1:numg)
4323        common/start1/deltatime,dz,comp
4324        common/holdupblok/vdr,vch,pch,chd,rh(1),bl(1:numg),bg(1:numg)
4325        common/flowrate/l(1:numg),g(1:numg)
4326        common/boundary/yboil(1:nmc),xref(1:nmc)
4327        common/parastudy/alfaky,alfakx,beta1,beta2
4328        common/frequ/yaztot,yazpro,option,decide,model,opmode,simpro
4329
4330        dimension sol(2*numg),yprime(2*numg),delta(2*numg)
4331
4332  c        equations for reboiler-column interface
4333        do 88 j=1,2*numg
4334          delta(j)=0.0d0
4335    88  continue
4336
4337        delta(1)=-3.0d0*sol(1)+4.0d0*sol(3)-sol(5)
4338
4339        delta(2)=sol(2)-yboil(comp)
4340
4341
4342  c       equations for column-condenser interface
4343
4344        delta(2*numg-1)= sol(numg*2-1)-xref(comp)
4345
4346        delta(2*numg)=sol((numg-2)*2)-4.0d0*sol((numg-1)*2)+
4347       +     3.0d0*sol(2*numg)
4348
4349
4350  c       equations for paked section
4351        do 12 j=2,numg-1
4352
4353          if(comp.eq.nmc) then
4354            flux=0.0d0
4355            do 77 kil=1,nmc-1
4356              coff=ky(kil,j)*a(j)/
4357       +           (ky(kil,j)/kx(kil,j)*keqcons(kil,j)+1.0d0)
4358              flux=flux-coff*(keqcons(kil,j)*xmolfrac(kil,j)-
4359       +           ymolfrac(kil,j))
4360     77       continue
4361          else
4362            coff=ky(comp,j)*a(j)/
4363       +         (ky(comp,j)/kx(comp,j)*keqcons(comp,j)+1.0d0)
4364            flux=coff*(keqcons(comp,j)*sol(j*2-1)-sol(j*2))
4365          end if
4366
4367          delta(j*2-1)=bl(j)*yprime(j*2-1)-l(j)/(dz)*0.5d0*
4368       +       (sol((j+1)*2-1)-sol((j)*2-1)+
4369       +       xpre(comp,(j+1))-xpre(comp,j))+flux
4370
4371          delta(j*2)=bg(j)*yprime(j*2)+g(j)/(dz)*0.5d0*
4372       +       (sol((j)*2)-sol((j-1)*2)+
4373       +       ypre(comp,j)-ypre(comp,(j-1)))-flux
4374
4375          if(model.eq.2) then
4376            delta(j*2-1)=delta(j*2-1)
4377       +         -laxdis(j)*bl(j)*0.5d0/(dz**2.0d0)*
4378       +         ((sol((j+1)*2-1)-2.0d0*sol(j*2-1)+sol((j-1)*2-1))+
4379       +         (xpre(comp,(j+1))-2.0d0*xpre(comp,j)+
4380       +         xpre(comp,(j-1))))
```

```
4381
4382            delta(j*2)=delta(j*2)
4383       +         -gaxdis(j)*bg(j)*0.5d0/(dz**2.0d0)*
4384       +         ((sol((j+1)*2)-2.0d0*sol(j*2)+sol((j-1)*2))+
4385       +         (ypre(comp,(j+1))-2.0d0*ypre(comp,j)+
4386       +         ypre(comp,(j-1))))
4387            end if
4388
4389    12    continue
4390          ipar=10
4391          return
4392          end
4393
4394
4395          subroutine turev(y,yprime)
4396    c ***********************************************************************
4397    c     dynamic mass transfer equations written in  dassl code
4398    c ***********************************************************************
4399
4400          implicit double precision (a-z)
4401          integer numg,nmc,j,comp,i,kil
4402          integer yaztot,yazpro,option,decide,model,opmode,simpro
4403          include 'param.f'
4404          common/molfrac/xmolfrac(1:nmc,1:numg),ymolfrac(1:nmc,1:numg)
4405          common/disperblok/laxdis(1:numg),gaxdis(1:numg)
4406          common/mastran/kx(1:nmc,1:numg),ky(1:nmc,1:numg)
4407          common/propla/a(1:numg),keqcons(1:nmc,1:numg)
4408          common/premolfrac/xpre(1:nmc,1:numg),ypre(1:nmc,1:numg)
4409          common/start1/deltatime,dz,comp
4410          common/holdupblok/vdr,vch,pch,chd,rh(1),bl(1:numg),bg(1:numg)
4411          common/flowrate/l(1:numg),g(1:numg)
4412          common/boundary/yboil(1:nmc),xref(1:nmc)
4413          common/parastudy/alfaky,alfakx,beta1,beta2
4414          common/frequ/yaztot,yazpro,option,decide,model,opmode,simpro
4415
4416          dimension y(2*numg),yprime(2*numg),gval(2*numg),z(2*numg,2*numg)
4417
4418
4419          do 22 i=1,2*numg
4420            do 33 j=1,2*numg
4421                z(i,j)=0.0d0
4422    33        continue
4423            gval(i)=0.0d0
4424           yprime(i)=0.0d0
4425    22     continue
4426           z(1,1)=-3.0d0
4427           z(1,3)=4.0d0
4428           z(1,5)=-1.0d0
4429           z(2,2)=1.0d0
4430           z(2*numg-1,2*numg-1)=1.0d0
4431           z(2*numg,2*(numg-2))=1.0d0
4432           z(2*numg,2*(numg-1))=-4.0d0
4433          z(2*numg,2*numg)=3.0d0
4434    c       equations for paked section
4435          do 12 j=2,numg-1
4436
4437            if(comp.eq.nmc) then
4438              flux=0.0d0
4439              do 77 kil=1,nmc-1
4440                coff=ky(kil,j)*a(j)/
4441       +             (ky(kil,j)/kx(kil,j)*keqcons(kil,j)+1.0d0)
4442              flux=flux-coff*(keqcons(kil,j)*xmolfrac(kil,j)-
4443       +             ymolfrac(kil,j))
4444    77        continue
4445            else
4446              coff=ky(comp,j)*a(j)/
4447       +           (ky(comp,j)/kx(comp,j)*keqcons(comp,j)+1.0d0)
4448              flux=coff*(keqcons(comp,j)*y(j*2-1)-y(j*2))
4449            end if
4450
4451            z(j*2-1,j*2-1)=1.0d0
4452            z(j*2,j*2)=1.0d0
4453            gval(j*2-1)=(l(j)/(dz)*0.5d0*
4454       +         (y((j+1)*2-1)-y((j)*2-1)+
4455       +         xpre(comp,(j+1))-xpre(comp,j))-flux)/bl(j)
4456
4457              gval(j*2)=(-g(j)/(dz)*0.5d0*
4458       +         (y((j)*2)-y((j-1)*2)+
4459       +          ypre(comp,j)-ypre(comp,(j-1)))+flux)/bg(j)
4460
4461            if(model.eq.2) then
4462              gval(j*2-1)=gval(j*2-1)+(
4463       +           laxdis(j)*bl(j)*0.5d0/(dz**2.0d0)*
4464       +           ((y((j+1)*2-1)-2.0d0*y(j*2-1)+y((j-1)*2-1))+
4465       +           (xpre(comp,(j+1))-2.0d0*xpre(comp,j)+
4466       +           xpre(comp,(j-1)))))/bl(j)
4467
4468              gval(j*2)=gval(j*2)+
4469       +           (gaxdis(j)*bg(j)*0.5d0/(dz**2.0d0)*
4470       +           ((y((j+1)*2)-2.0d0*y(j*2)+y((j-1)*2))+
4471       +           (ypre(comp,(j+1))-2.0d0*ypre(comp,j)+
4472       +           ypre(comp,(j-1)))))/bg(j)
4473            end if
4474
4475    12    continue
4476          call gauss(2*numg,numg,z,gval,yprime)
4477
4478
4479          return
4480          end
4481
4482          subroutine dres2(t,y,yprime,delta,ires,rpar,ipar)
4483    c ***********************************************************************
4484    c     ddassl code for reboiler holdup calculation
4485    c ***********************************************************************
4486
4487          implicit double precision (a-z)
4488          integer ires,ipar,numg,nmc
4489          include 'param.f'
4490          common/flowrate/l(1:numg),g(1:numg)
4491          common/start3/colvol,charge,spa,vd,pst,dp,vboil,liqref,
4492         +     rx,ed,crosarea
4493          dimension y(1),yprime(1),delta(1)
4494          delta(1)=yprime(1)+crosarea*(g(1)-l(1))
4495          return
4496          end
4497
4498          subroutine dres3(t,y,yprime,delta,ires,rpar,ipar)
4499    c ***********************************************************************
4500    c       new reboiler composition
4501    c       y=xfeed(i)     yprime=dxfeed(i)/dt
4502    c ***********************************************************************
4503
4504          implicit double precision (a-z)
4505          integer ires,ipar,nmc,numg,comp
4506          include 'param.f'
4507          common/holdupblok/vr,vlc,prch,ch,rebhold(1),bl(1:numg),bg(1:numg)
4508          common/flowrate/l(1:numg),g(1:numg)
4509          common/start3/colvol,charge,spa,vd,pst,dp,vboil,liqref,
4510         +      rx,ed,crosarea
4511          common/molfrac/xmol(1:nmc,1:numg),ymol(1:nmc,1:numg)
4512          dimension y(1:nmc),yprime(1:nmc),delta(1:nmc)
4513          do 1 comp=1,nmc
4514            delta(comp)=yprime(comp)-(1.0d0/rebhold(1))*
4515       +       (l(1)*crosarea*xmol(comp,1)-g(1)*crosarea*ymol(comp,1)-
4516       +        y(comp)*crosarea*(l(1)-g(1)))
4517     1    continue
4518          return
4519          end
4520
4521          subroutine dres4(t,xref,xprime,delta,ires,rpar,ipar)
4522    c ***********************************************************************
4523    c       new condenser - reflux drum compositions
4524    c       x=xref(i)     xprime=dxref(i)/dt
4525    c ***********************************************************************
4526
```

```
      presbot=p(1)
      return
      end


      block data
      implicit double precision (a-z)
      common/presdrop/height,packfact,xpack(1:14),
     +     ypack(1:14,1:6),presdr(1:6)
      common/fugcons/aa0,aa1,aa2,aa3,aa4,aa5,aa6,aa7,aa8,aa9
      common/consbl/gamaa,gamab,coneq(19,4)

      data xpack/0.02d0,0.03d0,0.04d0,0.05d0,0.06d0,0.1d0,0.2d0,
     +     0.3d0,0.4d0,0.6d0,1.0d0,2.0d0,3.0d0,4.0d0/
      data ypack/0.15d0,0.148d0,0.146d0,0.143d0,0.142d0,0.14d0,
     +     0.139d0,0.136d0,0.133d0,0.129d0,0.115d0,0.088d0,
     +     0.071d0,0.058d0,0.47d0,0.45d0,0.442d0,0.44d0,0.43d0,
     +     0.42d0,0.37d0,0.34d0,0.31d0,0.26d0,0.2d0,0.12d0,
     +     0.085d0,0.065d0,1.1d0,0.88d0,0.78d0,0.76d0,0.73d0,
     +     0.72d0,0.66d0,0.57d0,0.52d0,0.42d0,0.28d0,0.151d0,
     +     0.1d0,0.075d0,1.9d0,1.56d0,1.45d0,1.4d0,1.37d0,1.27d0,
     +     1.04d0,0.9d0,0.75d0,0.58d0,0.38d0,0.195d0,0.12d0,0.085d0,
     +     3.25d0,2.85d0,2.65d0,2.42d0,2.35d0,1.98d0,1.5d0,1.2d0,
     +     0.98d0,0.75d0,0.46d0,0.22d0,0.135d0,0.09d0,4.0d0,
     +     3.3d0,3.0d0,2.8d0,2.7d0,2.2d0,1.7d0,1.35d0,1.1d0,
     +     0.8d0,0.5d0,0.23d0,0.14d0,0.1d0/
      data presdr/4.0d0,8.0d0,21.0d0,42.0d0,83.0d0,125.0d0/

c       constants for physical properties of system components

c^Iconstants using by srk equation of state
      data gamaa,gamab/0.4274802327d0,0.08664035d0/

c^Iconstants for use in the grayson-streed correlation of the
c^Iliquid fugacity coefficient for simple fluid
c^Icalculation of k values, holland 1981, page 540

      data aa0,aa1,aa2,aa3,aa4,aa5,aa6,aa7,aa8,aa9/2.05135d0,
     +     -2.10899d0,0.0d0,-0.19396d0,0.02282d0,0.08852d0,
     +     0.0d0,-0.00872d0,-0.00353d0,0.00203d0/
c       coneq constants

      data coneq/-0.0817d0,0.3274d0,-0.5014d0,0.3870d0,-0.1342d0,
     +     -0.0230d0,-0.0124d0,0.1625d0,-0.2135d0,0.08643d0,
     +     0.05626d0,-0.3518d0,0.6194d0,-0.3809d0,-21.0d0,55.174d0,
     +     -33.637d0,-28.10d0,23.277d0,0.0933d0,-0.3445d0,
     +     0.4042d0,-0.2083d0,0.05473d0,0.022d0,-0.003363d0,
     +     -0.0796d0,0.08546d0,-0.0217d0,0.01937d0,-0.03055d0,
     +     0.06310d0,0.0d0,-16.0d0,30.699d0,19.645d0,-81.305d0,
     +    47.031d0,0.0890d0,-0.4344d0,0.7915d0,-0.7654d0,0.3367d0,
     +     0.0674d0,-0.06109d0,0.06261d0,-0.2378d0,0.1665d0,
     +     -0.01393d0,-0.003456d0,-0.1611d0,0.0d0,-6.550d0,7.8027d0,
     +     15.344d0,-37.04d0,20.169d0,19*0.0d0/

      end
```

```
c     ******************************************************************
c     *                                                                *
c     *                          COMBLOK.F                             *
c     *                Common blocks of PACBACDIS.F                    *
c     *                  used as an include file                       *
c     ******************************************************************

      common/name/namecomp
      common/identcons/a1(1:nmc),b1(1:nmc),c1(1:nmc),
     +     d1(1:nmc),e1(1:nmc)
      common/mixmolw/lmixwme(1:numg),vmixwme(1:numg),wme(1:nmc)
      common/denblok/tden(1:nmc),lden(1:nmc),solpar(1:nmc)
      common/fugcons/aa0,aa1,aa2,aa3,aa4,aa5,aa6,aa7,aa8,aa9
      common/consbl/gamaa,gamab,coneq(19,4)
      common/redbl/tr(1:nmc,1:numg),tbr(1:nmc,1:numg),pr(1:nmc,1:numg)
      common/phis/tc(1:nmc),tbp(1:nmc),pc(1:nmc),w(1:nmc),vc(1:nmc)
      common/zblok/zvap(1:nmc,1:numg)
      common/conblok/bcon,ccon,chtr(1:nmc),compon
      common/cpblok/cpa(1:nmc),cpb(1:nmc),cpc(1:nmc),cpd(1:nmc)
      common/visbl/visa(1:nmc),visb(1:nmc)
      common/heatfirst/hod,hid,coildo,coildi,kw,heati,acoil,tempsteam
      common/heattwo/xfeed(1:nmc)
      common/fcnblk/tfilm,heato,u,tstill,q
      common/disperblok/laxdis(1:numg),gaxdis(1:numg)
      common/holdupblok/volrdrum,volconhold,preconhold,conhold,
     +     rebhold(1),liqhold(1:numg),gashold(1:numg)
      common/flowrate/liqrate(1:numg),gasrate(1:numg)
      common/mastran/liqmasstr(1:nmc,1:numg),gasmasstr(1:nmc,1:numg)
      common/enthalblk/mixhl(1:numg),mixhv(1:numg),mixhlpre(1:numg),
     +     mixhvpre(1:numg)
      common/prehold/blpre(1:numg),bgpre(1:numg)
      common/molfrac/xmolfrac(1:nmc,1:numg),ymolfrac(1:nmc,1:numg)
      common/premolfrac/xpre(1:nmc,1:numg),ypre(1:nmc,1:numg)
      common/propla/intarea(1:numg),keqcons(1:nmc,1:numg)
      common/start1/deltatime,deltaz,comp
      common/start2/vollhold,volghold
      common/presblok/prestop,presbot,p(1:numg)
      common/start3/colvol,charge,specarea,voidfrac,pacsurten,dp,vboil,
     +     liqref,relaxfac,eqdia,crosarea
      common/diffblk/mgdif(1:nmc,1:numg),mldif(1:nmc,1:numg)
      common/surfacbl/liqsur(1:nmc,1:numg),mlst(1:numg)
      common/pachold/ghold(1:numg),lhold(1:numg)
      common/frequ/yaztot,yazpro,option,decide,model,opmode,simpro
      common/tempblk/ttk(1:numg)
      common/interface/xint(1:nmc,1:numg),yint(1:nmc,1:numg)
      common/boundary/yboil(1:nmc),xref(1:nmc)
      common/filenames/moltotal,phytotal,flowtotal,
     +     errortotal,totalref
      common/massflow/lrate(1:numg),grate(1:numg)
      common/gasdenbl/gasden(1:nmc,1:numg),mgden(1:numg)
      common/liqdenbl/liqden(1:nmc,1:numg),mlden(1:numg)
      common/gasvisbl/gasvis(1:nmc,1:numg),mgvis(1:numg)
      common/liqvisbl/liqvis(1:nmc,1:numg),mlvis(1:numg)
      common/presdrop/height,packfact,xpack(1:14),ypack(1:14,1:6),
     +     presdr(1:6)
      common/parastudy/alfaky,alfakx,beta1,beta2
      common/preflow/lrpre(1:numg),grpre(1:numg)
```

```
c*****************************************************************
c*                                                               *
c*                          PARAM.F                              *
c*            Defines the dimensions of PACBACDIS.F              *
c*                    used as an include file                    *
c*            The following parameters should be modified for    *
c*            the problem at hand                                *
c*            nmc : number of components of the charge mixture   *
c*            numg: number of Finite Difference nodes taken      *
c*                  to represent the packed section              *
c*****************************************************************

      parameter(nmc=3,numg=35)
```